# UNIVERSIDAD DE GUAYAQUIL

FACULTAD DE CIENCIAS MATEMÁTICAS Y FÍSICAS

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

**“ANÁLISIS COMPARATIVO DE RESPALDO Y RECUPERACIÓN DE BASE DE DATOS LICENCIADA (ORACLE UTILIZANDO RMAN) VS OPEN SOURCE (MYSQL UTILIZANDO MYSQL ADMINISTRATOR)”**

## MANUAL TÉCNICO Y DE USUARIO

Previa a la obtención del Título de:

## INGENIERO EN SISTEMAS COMPUTACIONALES

AUTOR: INGRID EVELYN CEVALLOS CEVALLOS

TUTOR: ING. FREDDY BURGOS

GUAYAQUIL – ECUADOR

2014

# ÍNDICE GENERAL


PORTADA I

ÍNDICE GENERAL II

MANUAL TÉCNICO III

MANUAL DE USUARIO IV

# MANUAL TÉCNICO

## Introducción

El Objetivo de este manual es poder ayudar a entender la estructura y el funcionamiento de cada una de las herramientas utilizadas, de tal manera que los administradores de base de datos puedan ajustarlo a sus conocimientos.

# HERRAMIENTAS PARA LA IMPLEMENTACION DEL PROYECTO

## CONFIGURAR LA BASE DE DATOS EN MODO ARCHIVELOG

El modo ARCHIVELOG de Oracle es un mecanismo de protección ante fallos de disco implementado por Oracle. Protegerá la base de datos ante posibles fallos físicos de disco y también ante eliminaciones o modificaciones no deseadas de los datos. Con este mecanismo tendremos varias ventajas:

- Se podrán realizar copias de seguridad físicas online, copias de los ficheros físicos sin detener la base de datos.
- Se podrá recuperar la base de datos en un momento específico del tiempo, se podrá recuperar la base de datos a una hora y día indicados.

La única desventaja es que este modo puede ralentizar (mínimamente) la base de datos y requiere de más espacio en disco para alojar los redo log offline (archivos de reconstrucción fuera de línea).

En realidad Oracle implementó este modo de ARCHIVELOG cuando los discos duros eran más susceptibles de fallar. Hoy en día con los discos duros profesionales de que disponemos en niveles de RAID adecuados es bastante difícil que ocurran fallos, aunque no es imposible. Además, este modo, como ya hemos comentado, tiene varias ventajas por las que

merece la pena usarlo.

A continuación se muestra y se explica cómo activar en modo archivelog de la base de datos Oracle.

En primer lugar se creará los directorios donde se van a almacenar los ArchiveLogs.

Luego desde una terminal de Centos le doy clic en ('Aplicacion' - 'Terminal'):

**[root@fcevallos ~]# mkdir -p /opt/backups_oracle/archivelogs**

**[root@fcevallos ~]# mkdir -p /opt/backups_oracle/backupset**

**[root@fcevallos ~]# chown oracle.dba -R /opt/backups_oracle**

Luego me conecto como usuario Oracle que es al que le voy a activar el modo Archive Log (ARCHIVELOG), con el comando:

**[root@fcevallos ~]# su oracle**

**[oracle@fcevallos ~]$ sqlplus /nolog**

**SQL>connect /as sysdba;**

Si ejecuto este comando y la base de datos está en modo OPEN (abierta):

**SQL>alter database archivelog;**

Me dará el siguiente error:

**alter database archivelog**

*****

**ERROR at line 1:**

**ORA-01126: database must be mounted EXCLUSIVE and not open for this operation**

Indicando que para poder cambiar la base de datos Oracle a modo archivado (archive log) hay que iniciarla en modo "mount" (montado). Para ello hay que detenerla e iniciarla con los siguientes comandos:

**SQL>shutdown immediate;**

**Resultado comando:**

**Database closed.**

**Database dismounted.**

**ORACLE instance shut down.**

La inicio en modo "mount" con el comando:

**SQL>startup mount;**

Para cambiar a modo archivado ejecuto el siguiente comando:

**SQL>alter database archivelog;**

Resultado comando:

**Database altered.**

Vuelvo a iniciar la base de datos:

**SQL>alter database open;**

Resultado comando:

**Database altered.**

Para activar el archivado automático compruebo el valor del parámetro "log_archive_start", si está a "false" lo pondremos a "true". Para consultar el valor actual del parámetro ejecuto el siguiente comando:

**SQL>show parameter log_archive_start;**

Devolverá el siguiente resultado:

**NAME TYPE VALUE**

**---------- -------- --------**

**log_archive_startboolean FALSE**

Voy a modificar el parámetro "log_archive_start" a "true" con el siguiente comando:

**SQL>alter system set LOG_ARCHIVE_START=TRUE SCOPE=spfile;**

Resultado comando:

**System altered.**

Al ejecutar "SCOPE=spfile" haremos que los cambios se guarden definitivamente.

Para que los cambios tengan efecto es recomendable parar y volver a iniciar la base de datos:

**SQL>shutdown immediate;**

Inicio la base de datos en modo normal:

**SQL>startup;**

Para comprobar en qué modo está la base de datos ejecuto el siguiente comando:

**SQL>select log_mode from v$database;**

Resultado:

**LOG_MODE**

**----------**

**ARCHIVELOG**(está en modo archivado)

**Nota:** Hay que activar el modo archivelog a todas las bases de datos que quiero respaldar.

## CREAR CATALOGO RMAN

El Catalogo de RMAN, se basa en crear un repositorio de información, un tablespace con un usuario y hacer que allí se guarde toda la información para gestionar las copias de seguridad de una base de datos.

Si RMAN no dispone de un catálogo de recuperación, toda la información será guardada en el ControlFile de la base de datos a la cual se conecta.

A continuación se muestra como crear un catálogo de rman.

En este esquema  le voy a crear una instancia distinta a la que quiero respaldar, en este caso creo previamente la BD con el nombre de catrman y el esquema de user_rman.

En la BD catalogo_2, nos conectamos vía SQL*Plus y vamos a crear un tablespace llamado "tbs_rman "

**[root@fcevallos ~]# su oracle**

**[oracle@fcevallos root]$ export ORACLE_SID=catrman**

**[oracle@fcevallos root]$ sqlplus / as sysdba**

SQL*Plus: Release 10.2.0.1.0 - Production on Jue Sep 13 16:20:55 2012

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

Conectado a una instancia inactiva.

**SQL> startup;**

Instancia ORACLE iniciada.

Total System Global Area 1191182336 bytes

Fixed Size 1219064 bytes

Variable Size 318768648 bytes

Database Buffers 855638016 bytes

Redo Buffers 15556608 bytes

Base de datos montada.

Base de datos abierta.

**SQL> select name from v$database;**

NAME

---

CATALOGO

En la base catrman, voy a crear el esquema user_rman, así como definir el tablespace, asignarle una cuota indefinida y por último le asigno los permisos necesarios al esquema user_rman:

**SQL> create tablespace tbs_rman**

**2  datafile '/opt/RMAN/TBS_RMAN.DBF'**

**3  SIZE 20M**

**4  extent management local uniform size 128k;**

Tablespace creado.

**SQL> create user user_rman identified by clave_rman**

**2  default tablespace tbs_rman**

**3  quota unlimited on tbs_rman;**

Usuario creado.

**SQL> GRANT RECOVERY_CATALOG_OWNER TO user_rman;**

Concesión terminada correctamente.

**SQL> GRANT CONNECT, RESOURCE TO user_rman;**

Concesión terminada correctamente.

**SQL>exit;**

Una vez que creo el esquema user_rman, me salgo del ambiente de SQL*Plus y me conecto con el cliente de RMAN al esquema que acabo de crear, mencionando que este es un catálogo y voy a crear el catalogo en este esquema.

**[root@fcevallos ~]# su oracle**

**[oracle@fcevallos root]$ cd $ORACLE_HOME/bin**

**[oracle@fcevallos bin]$ ls rman**

**rman**

**[oracle@fcevallos bin]$ export ORACLE_SID=catrman**

**[oracle@fcevallos bin]$ rman catalog user_rman/clave_rman**

Recovery Manager : Release 10.2.0.1.0 - Production on Jue Sep 13 16:40:58 2012

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

conectado a la base de datos del catálogo de recuperación

**RMAN>create catalog;**

catálogo de recuperación creado

**RMAN>exit;**

Lo último que me falta de hacer es registrar las base de datos creadas (**finandb1, finandb3, finandb4)** en el catálogo que acabo de crear, esto lo hago desde el cliente de rman, conectándome a la BD que quiero respaldar y al catálogo que voy a utilizar.

**[root@fcevallos ~]# su oracle**

**[oracle@fcevallos root]$ cd $ORACLE_HOME/bin**

**[oracle@fcevallos bin]$ export ORACLE_SID=catrman**

**[oracle@fcevallos bin]$ rman target sys/oracle@finandb1**

Recovery Manager : Release 10.2.0.1.0 - Production on Jue Sep 13 16:45:43 2012

Copyright (c) 1982, 2005, Oracle.  All rights reserved.

conectado a la base de datos destino: FINANDB1 (DBID=2960262359)

**RMAN> connect catalog user_rman/clave_rman**

conectado a la base de datos del catálogo de recuperación

**RMAN>register database;**

Base de datos registrada en el catálogo de recuperación

Iniciando resincronización completa del catálogo de recuperación

resincronización completa terminada

**RMAN>exit;**

**NOTA:** Este procedimiento lo hago para el resto de las demás bases de datos.

## BACKUP CON RMAN

En este punto muestro como realizar un backup de una base de datos con RMAN, para el cual diseñé un shell que me permita ejecutarlo de manera más rápida y sencilla vía comando:

**Nota:** El Shell está diseñado para cada una de las bases creadas.

En la línea **ORACLE_SID** se especificaran el nombre de todas bases a respaldar.

**----------BACKUP_grande.sh----------**

```
#!/bin/ksh

ORACLE_HOME=/u01/app/oracle/product/10.2.0/db_1;
export ORACLE_HOME
ORACLE_SID=finandb1;
export ORACLE_SID
PATH=$PATH:$ORACLE_HOME/bin;
export PATH
LOG=/opt/backups_oracle/archivelogs/logfullbackup_`date
+%d%m%Y_%I%p`.log;
export LOG
USUARIO_DBA=sys;
export USUARIO_DBA
CLAVE_DBA=cevallos;
export CLAVE_DBA
USUARIO_CAT=user_rman;
```

```
export USUARIO_CAT

CLAVE_CAT=clave_rman;

export CLAVE_CAT


echo "Conexión a RMAN para respaldar base de datos producción" >>
$LOG

echo " " >>$LOG

echo " " >>$LOG

echo "Log generado por RMAN:" >>$LOG

echo " " >>$LOG

echo " " >>$LOG


$ORACLE_HOME/bin/rman target=$USUARIO_DBA/$CLAVE_DBA
catalog=$USUARIO_CAT/$CLAVE_CAT@catalogo_2 << EOF >>$LOG

run {

allocate channel C1 device type DISK format
'/opt/backups_oracle/backupset/FULL_%d_%u_%t.bak';

backup database include current controlfile plus archivelog delete all input;
```

```
delete no promp to bsolete;

}

exit;

EOF



echo " " >>$LOG

echo " " >>$LOG

echo "Fin de ejecución de respaldo de archivelog con RMAN" >>$LOG

echo "********************************** " >>$LOG
```

## RECUPERAR BASE DE DATOS CON RMAN

Al igual que el backup de rman también he diseñado un shell que me permita ejecutarlo más rápido, vía comando:

**Nota:** El shell está diseñado para cada una de las bases creadas.

En la línea **ORACLE_SID** se especificarán todas bases a recuperar.

**----------Restore_grande.sh----------**

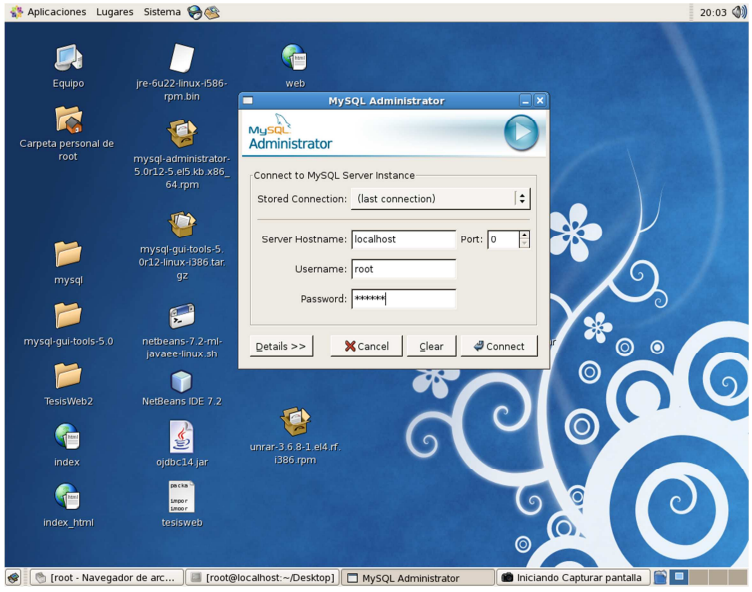
```
#!/bin/ksh
ORACLE_HOME=/u01/app/oracle/product/10.2.0/db_1;
export ORACLE_HOME
ORACLE_SID=finandb1;
export ORACLE_SID
PATH=$PATH:$ORACLE_HOME/bin;
export PATH
LOG=/opt/backups_oracle/archivelogs/restore_DB_`date
+%d%m%Y_%I%p`.log;
export LOG
USUARIO_DBA=sys;
export USUARIO_DBA
CLAVE_DBA=cevallos;
export CLAVE_DBA
USUARIO_CAT=user_rman;
export USUARIO_CAT
CLAVE_CAT=clave_rman;
export CLAVE_CAT
echo "Conexión a RMAN para restaurar y recuperar la base de datos
producción" >> $LOG
echo " " >>$LOG
echo " " >>$LOG
echo "Log generado por RMAN:" >>$LOG
echo " " >>$LOG
echo " " >>$LOG

$ORACLE_HOME/bin/sqlplus -s $USUARIO_DBA"/"$CLAVE_DBA " as
```

```
sysdba" << EOF >>$LOG
shutdown immediate;
startup mount;
exit
EOF

$ORACLE_HOME/bin/rman target=$USUARIO_DBA/$CLAVE_DBA
catalog=$USUARIO_CAT/$CLAVE_CAT@catalogo_2 << EOF >>$LOG
RESTORE DATABASE;
RECOVER DATABASE;
EXIT;
EOF

$ORACLE_HOME/bin/sqlplus -s $USUARIO_DBA"/"$CLAVE_DBA "
assysdba" << EOF >>$LOG
ALTER DATABASE OPEN;
SHUTDOWN IMMEDIATE;
STARTUP;
exit
EOF
echo " " >>$LOG
echo " " >>$LOG
echo "Fin de ejecución de restauración de la base de datos de producción
con RMAN" >>$LOG
echo "********************************** " >>$LOG
```

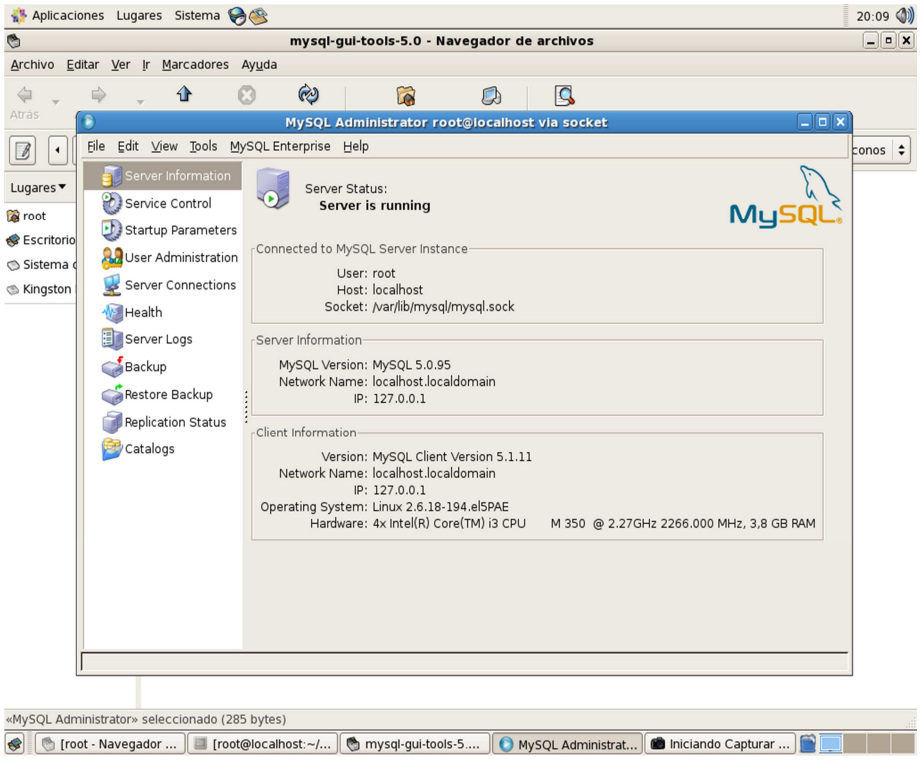
## CONFIGURACION DE MYSQL ADMINISTRATOR

En primer lugar se debe configurar la primera conexión con el servidor de MySQL para poder acceder a la administración del mismo.

En la pantalla de configuración se deberán especificar los datos del servidor, nombre de usuario, contraseña y puerto de la base de datos.




Una vez introducidos de forma correcta, aparece la pantalla principal del programa:




Desde la pantalla principal se puede ver la información relativa al servidor de base datos al que me he conectado, así como las características de la máquina en la que se está ejecutando el programa.

En la barra de menú a través de Tools, se tiene acceso a una consola del sistema Linux, un cliente de conexión a MySQL bajo línea de comandos y a MySQLQuery Browser. Desde este mismo menú, se accede a las opciones de configuración del administrador y de las conexiones que hay configuradas.

## RESPALDO DE BASE DE DATOS CON MYSQLDUMP

Los respaldos son una de las tareas más importante dentro de la administración de una base de datos.

La manera más simple de hacer un respaldo con MySQL es usar el comando mysqldump. Este retorna un fichero con instrucciones SQL para generar las tablas de la base de datos y rellenarlas con ls información que contenía en el momento de ejecutar el mysqldump. Este sistema es lento, pero ofrece el máximo de compatibilidad a la hora de hacer una migración.

Al igual que el backup con Oracle también he diseñado un shell para Mysql que me permita ejecutarlo más rápido, vía comando:

**Nota:** El shell está diseñado para cada una de las bases creadas.

**----------backup_grande.sh----------**

```
#!/bin/ksh

echo Iniciando Respaldo en Base de Datos Mysql

mysqldump -u root –pcentos zeusaudit >
/opt/backup_mysql/zeusaudit.sql

echo Fin de Respaldo de Base de Datos.
```

Donde las opciones son como se detallan a continuación:

**-u:** El nombre de usuario que se utilizará cuando se conecte con el servidor, el valorpredeterminado es el del usuario actual.

**-p:** La contraseña a usar al conectar con el servidor, no puede haber un espacio entre la opción y la contraseña.

**zeusaudit:** es el nombre de la base de datos.

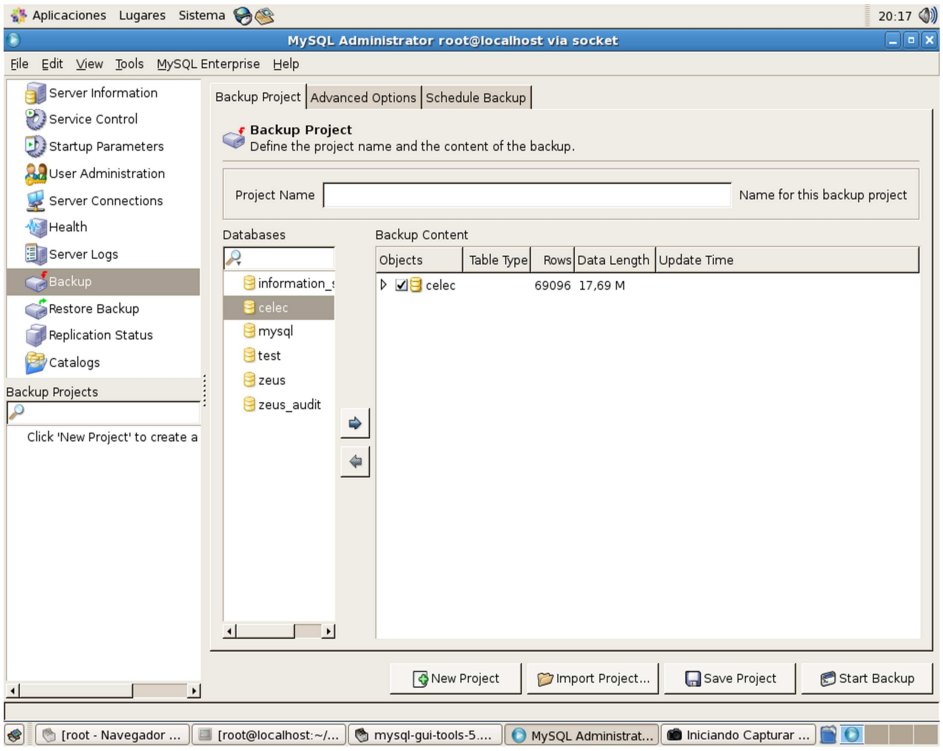
## RESPALDO DE BASE DE DATOS CON MYSQL ADMINISTRATOR

Para hacer un respaldo desde un entorno gráfico, debemos seguir los pasos siguientes.

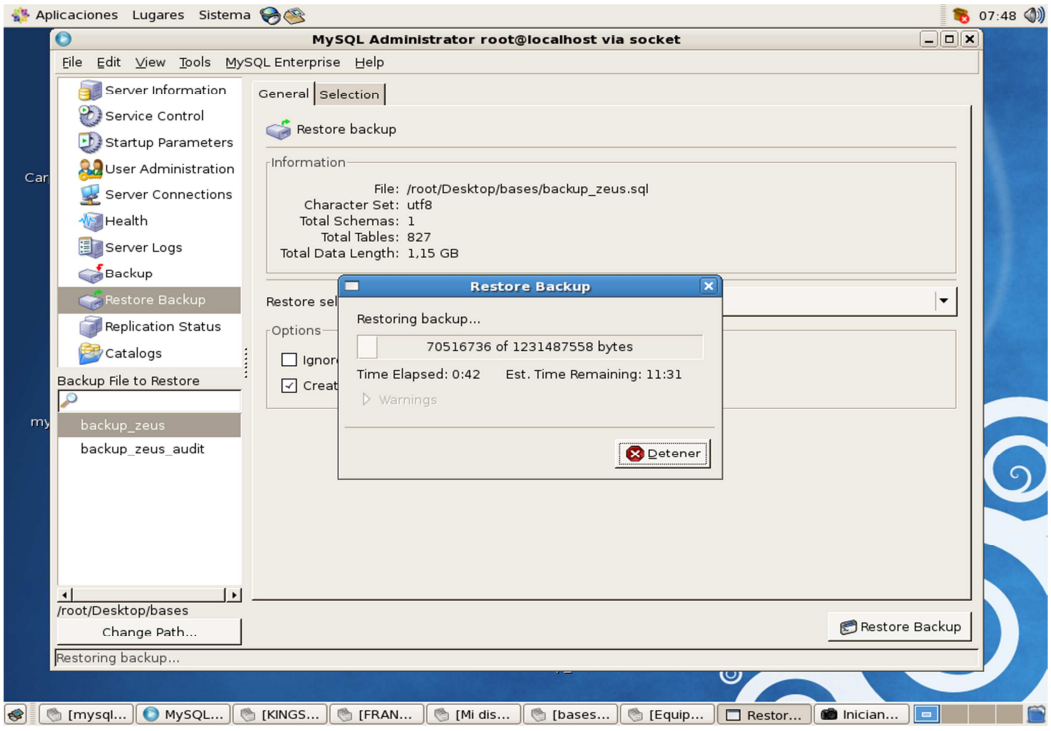
1.- Ingreso al entorno gráfico Mysql Administrator.

2.-A continuación se abre una ventana, en la parte izquierda de la misma damos click sobre la opción **Backup** y a continuación sobre el botón **New Project**.

3.- Luego coloco un nombre al proyecto y selecciono la base de datos de la que quiero hacer el respaldo y pulso el botón que tiene una flechita.




4.- Una vez que haya seleccionado la base a respaldar doy click en el botón **Start Backup**, al hacerlo me aparece un cuadro de dialogo donde busco una ubicación para el archivo de backup (.sql), luego se ejecutará el backup y mostrará una ventana notificando que se realizó de manera correcta.




## RECUPERACION DE BASE DE DATOS CON MYSQLDUMP

Si tenemos un fichero backup.sql que contiene una sola base de datos, la podemos restaurar así:

mysql -u root -p nombre_db<backup.sql

La base de datos nombre_bd debe existir. Si queremos restaurar varias bases de datos que están en un solo fichero, este contendrá los comandos CREATE DATABASE correspondientes. Simplemente hay que hacer:

mysql -u root -p <backup.sql

Para realizar la recuperación de la base de datos en Mysql también he

diseñado un shell para Mysql que me permita ejecutarlo más rápido, vía comando:

**Nota:** El shell está diseñado para cada una de las bases creadas.

**----------restore_grande.sh----------**

```
#!/bin/ksh
echo Iniciando Recuperación en Base de Datos Mysql
mysql -u root –pcentos zeusaudit< /opt/backups_mysql/zeusaudit.sql
echo Fin de Recuperación de Base de Datos.
```

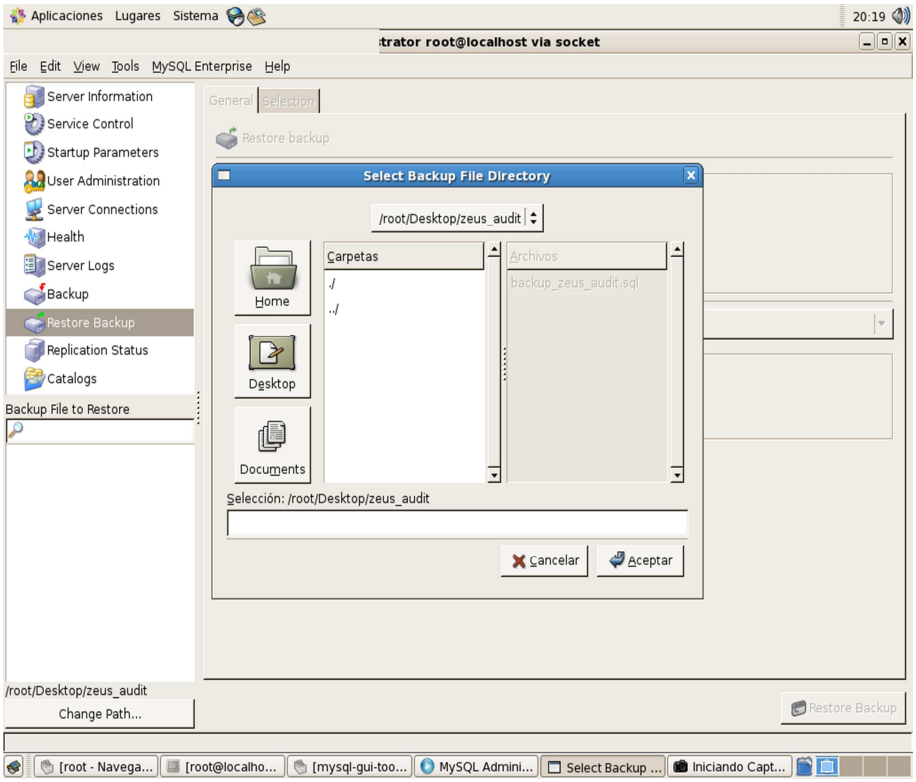
## RECUPERACION DE BASE DE DATOS CON MYSQL ADMINISTRATOR

Para hacer la recuperación de la base de datos desde un entorno gráfico, debemos seguir los pasos siguientes.

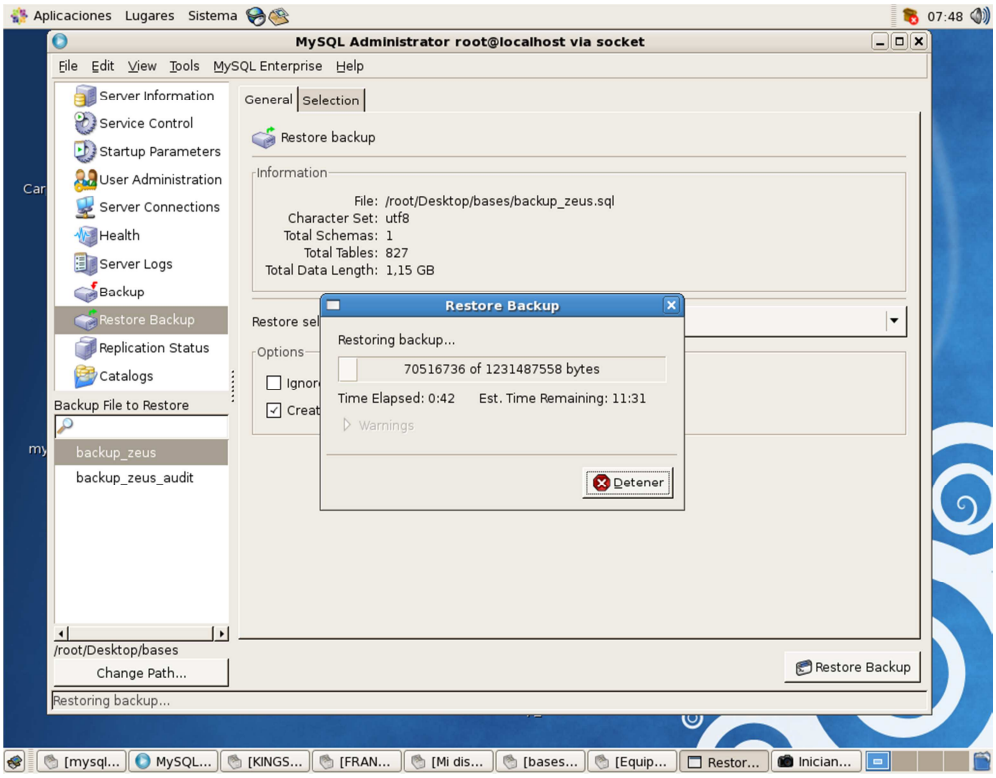
1.- Ingreso al entorno gráfico Mysql Administrator.

2.-A continuación se abre una ventana, en la parte izquierda de la misma doy click

sobre la opción **Restore Backup** y luego hago clic sobre la opción **Change Path** para buscar la ubicación del backup que quiero restaurar.




**3.-** Luego de seleccionar el archivo, busco la base donde quiero restaurar y pulso sobre el botón **Restore Backup**. Si todo se ejecuta de manera correcta se mostrara una ventana como la que aparece a continuación y pulso **Close**.




**4.-** En la pestaña **Selection**, se puede ver el contenido de que fue recuperado.

## PUTTY

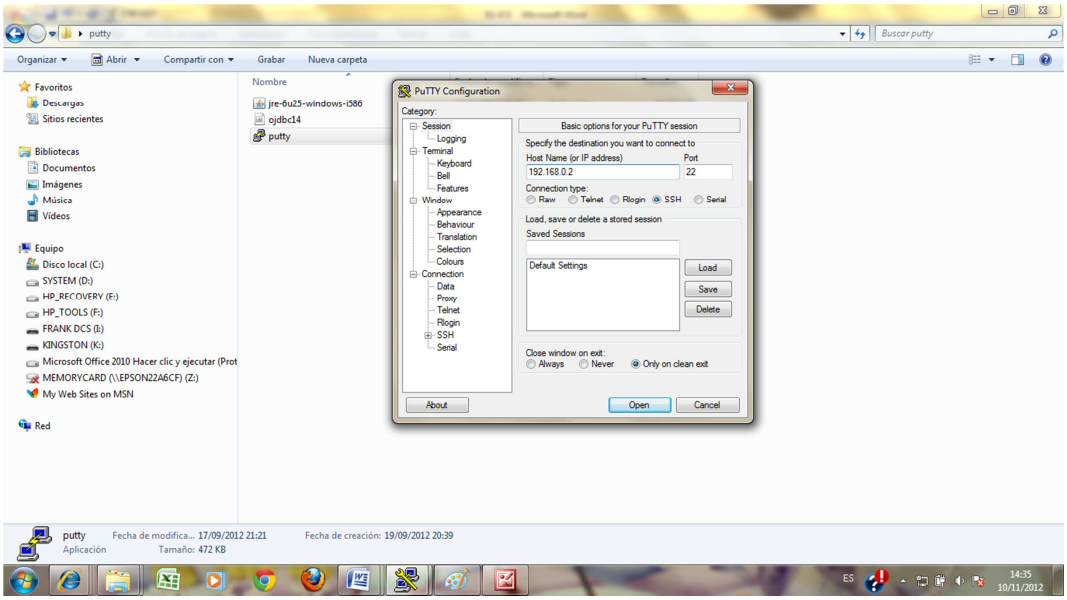
Es un cliente de red que soporta los protocolos SSH, Telnet y R login y sirve principalmente para iniciar una sesión remota con otra máquina o servidor. Es de licencia libre y a pesar de su sencillez es muy funcional y configurable.

Este programa me permitirá trabajar desde el servidor de la aplicación sin

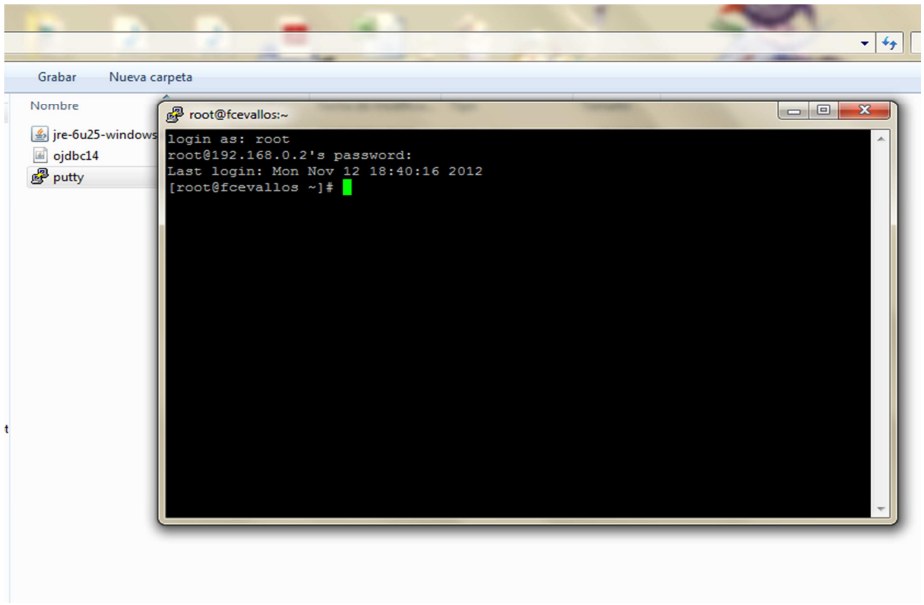
tener que hacerlo en cada uno de los servidores de gestores de base de datos.

A continuación voy a conectarme por medio del protocolo SSH con putty, para ejecutarlo no es necesario instalarlo, se ejecuta directamente.

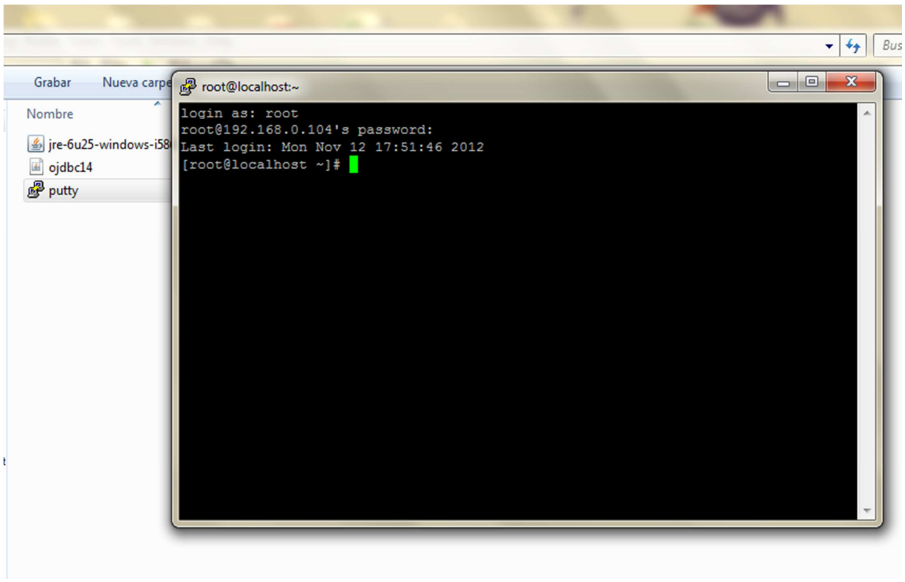
En el menú de configuración selecciono la categoría **Session** e ingreso la IP en el campo **Host Name** del servidor al que quiero conectarme en este caso estoy ingresando la IP del servidor de Oracle (192.168.0.110), selecciono la categoría SSH y por ultimo doy clic en el botón **Open**.




Al iniciar la conexión, se abrirá la ventana del terminal. Ingreso su nombre de usuario y pulso enter, después ingreso su contraseña y pulse enter. Si el nombre de usuario y password son correctos podrá iniciar la sesión SSH.




**Conexión remota a Oracle**




**Conexión remota a Mysql**

Y listo ya estoy conectada al servidor donde está instalado Oracle. De igual manera realizo esta conexión pero para el servidor donde está instalado Mysql con su respectiva IP 192.168.0.109.

# DETALLE DE TABLAS

Las estructuras creadas para el desarrollo del presente proyecto son:

**Tabla Rol**

Esta tabla registra los tipos de perfiles que pueden ingresar al aplicativo.

| Nombre | Tipo Dato | Descripción |
|---|---|---|
| **Id** | Int | Registra el código del tipo de rol. |
| **Nombre** | nchar(50) | Nombre del tipo de rol. |

**Tabla Usuario**

La creación de esta tabla es para registrar al usuario que interactuara con el aplicativo.

| Nombre | Tipo Dato | Descripción |
|---|---|---|
| **Id** | Int | Código del usuario. |
| **Nombre** | nvarchar(50) | Registra el nombre del usuario. |
| **Estado** | nvarchar(50) | muestra el nombre del gerente de la empresa. |
| **Clave** | nvarchar(13) | Registra la clave del usuario. |

**Tabla LogBase**

Esta tabla muestra el registro de los respaldos y restauración realizados.

| Nombre | Tipo Dato | Descripción |
|---|---|---|
| **Id** | Int | Código de LogBase |
| **Fecha** | nvarchar(50) | Muestra la fecha en que se realiza el respaldo y recuperación de las bases de datos. |
| **Oracle** | nvarchar(50) | Muestra el tiempo que duró el respaldo. |
| **MySql** | nvarchar(13) | Muestra el tiempo que duró el respaldo. |

## CODIGO FUENTE

```
package ec.com.tesis;


import com.jcraft.jsch.ChannelExec;

import com.jcraft.jsch.JSch;

import com.jcraft.jsch.JSchException;

import com.jcraft.jsch.Session;

import com.jcraft.jsch.UserInfo;

import ec.com.tesis.conexion.BaseMysql;

import ec.com.tesis.conexion.BaseOracle;

import ec.com.tesis.conexion.MostrarBases;

import ec.com.tesis.conexion.MostrarBasesOracle;

import ec.com.tesis.entity.LogBase;
```

```
import ec.com.tesis.sesion.LogBaseFacadeLocal;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.Serializable;
import java.text.Format;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.application.FacesMessage;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import org.primefaces.context.RequestContext;
import org.primefaces.event.SelectEvent;
import org.primefaces.model.DefaultTreeNode;
```

```
import org.primefaces.model.TreeNode;
import org.primefaces.model.chart.CartesianChartModel;
import org.primefaces.model.chart.ChartSeries;
import org.primefaces.model.chart.PieChartModel;

/**
 *
 * @author icevallos
 */
//@ManagedBean(name="testBean")
//@ViewScoped
public class TestBean implements Serializable {
//

    private String logmysql;
    private String logoracle;
    private long tiempomyql;
    private long tiemporacle;
    private boolean vergrafico = false;
    private String iniciomysql;
    private String finmysql;
    private String finoracle;
    private List<Resultados> listResult;
```

```
private String base;
private String baseMysqlSeleted;
private String baseOracleSeleted;
private int tipo;
private TreeNode root;
String conexion_mysql;
String conexion_oracle;
TreeNode mysql;
TreeNode oracle;
TreeNode mysqlRestore;
TreeNode oracleRestore;
TreeNode mysqlBackup;
TreeNode oracleBackup;
private List<Recurso> caracteristicas;
private String procesador_mysql;
private String memoria_mysql;
private String disco_mysql;
private String procesador_oracle;
private String memoria_oracle;
private String disco_oracle;
private String log_mysql;
private CartesianChartModel restore;
private List<Recurso> log;
```

```
private PieChartModel pieModel;
private String tituloGrafico;
private long max = 0;
private String claveOracle;
@EJB
private LogBaseFacadeLocal logBaseFacadeLocal;
private LogBase logbase;
private List<LogBase> listLogBase;
private PieChartModel grafico;
private MostrarBases baseM;
private MostrarBasesOracle baseOracle;
private List<Resultados> resul;
private List<Resultados> resulOracle;
private String mostrarLogOracle;
private List<Object[]> tablaBase;
private List<Object[]> tablaoracle;
/**
 * Creates a new instance of TestBean
 */
@PostConstruct
public void iniciar() {

    //createMeterGaugeModel();
```

```
        root = new DefaultTreeNode("root", null);
        mysql = new DefaultTreeNode(new Resultados("Mysql", "-", "-", "-"),
root);
        oracle = new DefaultTreeNode(new Resultados("Oracle", "-", "-", "-"),
root);
        mysqlRestore = new DefaultTreeNode(new Resultados("Restore", "-
", "-", "-"), mysql);
        oracleRestore = new DefaultTreeNode(new Resultados("Restore", "-
", "-", "-"), oracle);
        mysqlBackup = new DefaultTreeNode(new Resultados("Backup", "-",
"-", "-"), mysql);
        oracleBackup = new DefaultTreeNode(new Resultados("Backup", "-",
"-", "-"), oracle);
        //getMemoriaMyql();
        //getMemoriaOracle();
        caracteristicas = new ArrayList<Recurso>();
//        caracteristicas.add(new Recurso("Procesador", procesador_mysql,
procesador_oracle));
//        caracteristicas.add(new Recurso("Memoria", memoria_mysql,
memoria_oracle));
//        caracteristicas.add(new Recurso("Disco", disco_mysql,
disco_oracle));
```

```
        caracteristicas.add(new Recurso("Procesador", "Intel(R) Core(TM) i3
CPU  M 350  @ 2.27GHz", "Intel(R) Core(TM) i3 CPU  M 350  @
2.27GHz"));
        caracteristicas.add(new Recurso("Memoria", "3945056 kB", "3938740
kB"));
        caracteristicas.add(new Recurso("Disco", "50 GB", "50 GB"));
        log = new ArrayList<Recurso>();
        log.add(new Recurso("Procesos", "", ""));
        log.add(new Recurso("Procesador", "", ""));
        log.add(new Recurso("Memoria", "", ""));
        log.add(new Recurso("Memoria Virtual", "", ""));
        createPieModel();
        verGrafico();

         grafico = new PieChartModel();

            grafico.set("Mysql",10);
            grafico.set("Oracle", 10);

        conexion_mysql = BaseMysql.getConexion("mysql");
        conexion_oracle = BaseOracle.getConexion("finandb");

        if (conexion_mysql.equals("OK")) {
```

```
    }
    //new VerRecursos().start();&& conexion_oracle.equals("OK")
    // ccc();
    //tablas();
}

public void ccc() {
    try {
        baseM = new MostrarBases();
        List<String> bases = baseM.buscarBases();
        for (String base1 : bases) {
            System.out.println(base1);
        }
    } catch (Exception e) {
        System.out.println(e.getMessage());
    }
}

public void tablas() {
    try {
        /* baseM = new MostrarBases();
         List<String> bases = baseM.mostrarTablas("celec");
```

```
            List<Object[]>  datos=baseM.totalDeTablasYRegistros(bases,
"celec");
            for (Object[] base1 : datos) {
            System.out.println(base1[0]);
            System.out.println(base1[1]);*/
           baseM = new MostrarBases();
         //   List<String> bases = baseM.mostrarTablas(baseMysqlSeleted);
          // List<Object[]> datos = baseM.totalDeTablasYRegistros(bases,
baseMysqlSeleted);
            //resul = new ArrayList<Resultados>();
            //for (Object[] base1 : datos) {
              //  resul.add(new Resultados((String) base1[0], "", "", (String)
base1[1]));

            //}
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }

    public void chooseCar() {
        RequestContext.getCurrentInstance().openDialog("selectLog");
    }
```

```
    public void chooseLog() {
        Map<String, Object> options = new HashMap<String, Object>();
        options.put("modal", true);
        options.put("draggable", false);
        options.put("resizable", false);
        options.put("contentHeight", 600);
        //hint: available options are modal, draggable, resizable, width,
height, contentWidth and contentHeight

        RequestContext.getCurrentInstance().openDialog("selectLogS",
options, null);
        //RequestContext.getCurrentInstance().openDialog("selectLogS");
    }

    public void onCarChosen(SelectEvent event) {
        //FacesMessage message = new
FacesMessage(FacesMessage.SEVERITY_INFO, "Car Selected",
"Model:" + car.getModel());
        //FacesContext.getCurrentInstance().addMessage(null, message);
    }

    public PieChartModel getPieModel() {
```

```
        return pieModel;
    }

    private void createPieModel() {
        pieModel = new PieChartModel();

        pieModel.set("Mysql", 0);
        pieModel.set("Oracle", 0);

    }
public void actionEvent(ActionEvent e) {
        Format formatter;
        Date fechaActual = new Date();
        formatter = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
        if (conexion_mysql.equals("OK")) {//&&
conexion_oracle.equals("OK")) {
            switch (tipo) {

                case 1:
                    tituloGrafico = "Restore";
                    if (base.equals("1")) {
                        fechaActual = new Date();
                        iniciomysql = formatter.format(fechaActual);
```

```
                runMYSQL("sh -x /opt/mysql/restore_pequeno.sh");
                fechaActual = new Date();
                finmysql = formatter.format(fechaActual);
                TreeNode mysqlRestorePequena = new
DefaultTreeNode(new Resultados("PequeÃ±a", iniciomysql, finmysql, "" +
tiempomyql), mysqlRestore);
                runOralce("sh -x /opt/oracle/restore_pequeno.sh");
                fechaActual = new Date();
                finoracle = formatter.format(fechaActual);
                TreeNode oracleRestorePequena = new
DefaultTreeNode(new Resultados("PequeÃ±a", finmysql, finoracle, "" +
tiemporacle), oracleRestore);
                pieModel = new PieChartModel();
                baseMysqlSeleted = "celec";
                baseOracleSeleted = "finandb4";
                claveOracle = "ecevallos";
                tiempomyql = 10;
//                tiemporacle = 30;
                if (tiempomyql > tiemporacle) {
                    max = tiempomyql;
                } else {
                    max = tiemporacle;
                }
```

```
                pieModel.set("Mysql", tiempomyql);
                pieModel.set("Oracle", tiemporacle);
                restore = new CartesianChartModel();
                ChartSeries oracle_restore = new ChartSeries();
                oracle_restore.setLabel("Oracle");
                oracle_restore.set("Tiempo (milseg)", tiemporacle);
                ChartSeries mysql_restore = new ChartSeries();
                mysql_restore.setLabel("Mysql");
                mysql_restore.set("Tiempo (milseg)", tiempomyql);
                restore.addSeries(mysql_restore);
                restore.addSeries(oracle_restore);
                LogBase lobBases = new LogBase();
                lobBases.setId(logBaseFacadeLocal.count() + 1);
                lobBases.setFecha(new Date());
                lobBases.setMysql(tiempomyql);
                lobBases.setOracle(tiemporacle);
                lobBases.setTipo("Restore");
                logBaseFacadeLocal.create(lobBases);
                SimpleDateFormat format = new
SimpleDateFormat("ddMMyyyy");
                mostrarLogOracle = "cat
/opt/restore_oracle/archivelogs/restore4_DB_" + format.format(new
Date()) + "*.log";
```

```
                } else if (base.equals("2")) {
                    fechaActual = new Date();
                    iniciomysql = formatter.format(fechaActual);
                    runMYSQL("sh -x /opt/mysql/restore_mediano.sh");
                    fechaActual = new Date();
                    finmysql = formatter.format(fechaActual);
                    TreeNode mysqlRestoreMedina = new
DefaultTreeNode(new Resultados("Mediana", iniciomysql, finmysql, "" +
tiempomyql), mysqlRestore);
                    runOralce("sh -x /opt/oracle/restore_mediano.sh");
                    //runOralce("cat
/opt/backups_oracle/archivelogs/log3fullbackup_24092012_10.log");
                    fechaActual = new Date();
                    baseMysqlSeleted = "zeus";
                    baseOracleSeleted = "finandb3";
                     claveOracle = "icevallos";
                    finoracle = formatter.format(fechaActual);
                    TreeNode oracleRestoreMediana = new
DefaultTreeNode(new Resultados("Mediana", finmysql, finoracle, "" +
tiemporacle), oracleRestore);
                    pieModel = new PieChartModel();
                    if (tiempomyql > tiemporacle) {
                        max = tiempomyql;
```

```
                } else {
                    max = tiemporacle;
                }
                pieModel.set("Mysql", tiempomyql);
                pieModel.set("Oracle", tiemporacle);
                restore = new CartesianChartModel();
                ChartSeries oracle_restore = new ChartSeries();
                oracle_restore.setLabel("Oracle");
                oracle_restore.set("Tiempo (milseg)", tiemporacle);
                ChartSeries mysql_restore = new ChartSeries();
                mysql_restore.setLabel("Mysql");
                mysql_restore.set("Tiempo (milseg)", tiempomyql);
                restore.addSeries(mysql_restore);
                restore.addSeries(oracle_restore);
                LogBase lobBases = new LogBase();
                lobBases.setId(logBaseFacadeLocal.count() + 1);
                lobBases.setFecha(new Date());
                lobBases.setMysql(tiempomyql);
                lobBases.setOracle(tiemporacle);
                lobBases.setTipo("Restore");
                logBaseFacadeLocal.create(lobBases);
                SimpleDateFormat format = new
SimpleDateFormat("ddMMyyyy");
```

```
                mostrarLogOracle = "cat
/opt/restore_oracle/archivelogs/restore3_DB_" + format.format(new
Date()) + "*.log";

            } else if (base.equals("3")) {
                baseMysqlSeleted = "zeus_audit";
                baseOracleSeleted = "finandb1";
                 claveOracle = "cevallos";
                iniciomysql = formatter.format(fechaActual);
                runMYSQL("sh -x /opt/mysql/restore_grande.sh");
                fechaActual = new Date();
                finmysql = formatter.format(fechaActual);
                TreeNode mysqlRestoreGrande = new
DefaultTreeNode(new Resultados("Grande", iniciomysql, finmysql, "" +
tiempomyql), mysqlRestore);
                runOralce("sh -x /opt/oracle/restore_grande.sh");
                finoracle = formatter.format(fechaActual);
                TreeNode oracleRestoreGrande = new
DefaultTreeNode(new Resultados("Grande", finmysql, finoracle, "" +
tiemporacle), oracleRestore);
                pieModel = new PieChartModel();
                if (tiempomyql > tiemporacle) {
                    max = tiempomyql;
```

```
                } else {
                    max = tiemporacle;
                }
                pieModel.set("Mysql", tiempomyql);
                pieModel.set("Oracle", tiemporacle);
                restore = new CartesianChartModel();
                ChartSeries oracle_restore = new ChartSeries();
                oracle_restore.setLabel("Oracle");
                oracle_restore.set("Tiempo (seg)", tiemporacle);
                ChartSeries mysql_restore = new ChartSeries();
                mysql_restore.setLabel("Mysql");
                mysql_restore.set("Tiempo (seg)", tiempomyql);
                restore.addSeries(mysql_restore);
                restore.addSeries(oracle_restore);
                SimpleDateFormat format = new
SimpleDateFormat("ddMMyyyy");
                mostrarLogOracle = "cat
/opt/restore_oracle/archivelogs/restore1_DB_" + format.format(new
Date()) + "*.log";

                LogBase lobBases = new LogBase();
                lobBases.setId(logBaseFacadeLocal.count() + 1);
                lobBases.setFecha(new Date());
```

```
                lobBases.setMysql(tiempomyql);
                lobBases.setOracle(tiemporacle);
                lobBases.setTipo("Restore");
                logBaseFacadeLocal.create(lobBases);
            }

            //Restaurar
            break;
        case 2:
            tituloGrafico = "Backup";
            if (base.equals("1")) {
                iniciomysql = formatter.format(fechaActual);
                runMYSQL("sh -x /opt/mysql/backup_pequeno.sh");
                baseMysqlSeleted = "celec";
                baseOracleSeleted = "finandb4";
                 claveOracle = "ecevallos";
                fechaActual = new Date();
                finmysql = formatter.format(fechaActual);
                TreeNode mysqlBackupPequena = new
DefaultTreeNode(new Resultados("PequeÃ±a", iniciomysql, finmysql, "" +
tiempomyql), mysqlBackup);
                logoracle = "Procesando respaldo";
                runOralce("sh -x /opt/oracle/backup_pequeno.sh");
```

```
                logoracle = "Fin respaldo";
                fechaActual = new Date();
                finoracle = formatter.format(fechaActual);

                SimpleDateFormat format = new
SimpleDateFormat("ddMMyyyy");
                mostrarLogOracle = "cat
/opt/backups_oracle/archivelogs/log4fullbackup_" + format.format(new
Date()) + "*.log";
                TreeNode oracleBackupPequena = new
DefaultTreeNode(new Resultados("PequeÃ±a", finmysql, finoracle, "" +
tiemporacle), oracleBackup);
                pieModel = new PieChartModel();
                if (tiempomyql > tiemporacle) {
                    max = tiempomyql;
                } else {
                    max = tiemporacle;
                }
                pieModel.set("Mysql", tiempomyql);
                pieModel.set("Oracle", tiemporacle);
                restore = new CartesianChartModel();
                ChartSeries oracle_restore = new ChartSeries();
                oracle_restore.setLabel("Oracle");
```

```
            oracle_restore.set("Tiempo (milseg)", tiemporacle);
            ChartSeries mysql_restore = new ChartSeries();
            mysql_restore.setLabel("Mysql");
            mysql_restore.set("Tiempo (milseg)", tiempomyql);
            restore.addSeries(mysql_restore);
            restore.addSeries(oracle_restore);

            LogBase lobBases = new LogBase();
            lobBases.setId(logBaseFacadeLocal.count() + 1);
            lobBases.setFecha(new Date());
            lobBases.setMysql(tiempomyql);
            lobBases.setOracle(tiemporacle);
            lobBases.setTipo("Backup");

            logBaseFacadeLocal.create(lobBases);
        } else if (base.equals("2")) {
            baseMysqlSeleted = "zeus";
            baseOracleSeleted = "finandb3";
            claveOracle = "icevallos";
            iniciomysql = formatter.format(fechaActual);
            runMYSQL("sh -x /opt/mysql/backup_mediano.sh");
            fechaActual = new Date();
            finmysql = formatter.format(fechaActual);
```

```
                TreeNode mysqlBackupMedina = new
DefaultTreeNode(new Resultados("Mediana", iniciomysql, finmysql, "" +
tiempomyql), mysqlBackup);
                runOralce("sh -x /opt/oracle/backup_mediano.sh");
                fechaActual = new Date();
                finoracle = formatter.format(fechaActual);
                // mostrarLogOracle="log4fullbackup_21092013_01.log";

                SimpleDateFormat format = new
SimpleDateFormat("ddMMyyyy");
                mostrarLogOracle = "cat
/opt/backups_oracle/archivelogs/log3fullbackup_" + format.format(new
Date()) + "*.log";

//                  //runOralce("cat
/opt/backups_oracle/archivelogs/log3fullbackup_24092012_10.log");
                TreeNode oracleBackupMediana = new
DefaultTreeNode(new Resultados("Mediana", finmysql, finoracle, "" +
tiemporacle), oracleBackup);
                pieModel = new PieChartModel();
                if (tiempomyql > tiemporacle) {
                    max = tiempomyql;
                } else {
```

```
            max = tiemporacle;
        }
        pieModel.set("Mysql", tiempomyql);
        pieModel.set("Oracle", tiemporacle);
        restore = new CartesianChartModel();
        ChartSeries oracle_restore = new ChartSeries();
        oracle_restore.setLabel("Oracle");
        oracle_restore.set("Tiempo (seg)", tiemporacle);
        ChartSeries mysql_restore = new ChartSeries();
        mysql_restore.setLabel("Mysql");
        mysql_restore.set("Tiempo (seg)", tiempomyql);
        restore.addSeries(mysql_restore);
        restore.addSeries(oracle_restore);

        LogBase lobBases = new LogBase();
        lobBases.setId(logBaseFacadeLocal.count() + 1);
        lobBases.setFecha(new Date());
        lobBases.setMysql(tiempomyql);
        lobBases.setOracle(tiemporacle);
        lobBases.setTipo("Backup");
        logBaseFacadeLocal.create(lobBases);
    } else if (base.equals("3")) {
        baseMysqlSeleted = "zeus_audit";
```

```
                baseOracleSeleted = "finandb1";
                 claveOracle = "cevallos";
                iniciomysql = formatter.format(fechaActual);
                runMYSQL("sh -x /opt/mysql/backup_grande.sh");
                fechaActual = new Date();
                finmysql = formatter.format(fechaActual);
                TreeNode mysqlBackupGrande = new
DefaultTreeNode(new Resultados("Grande", iniciomysql, finmysql, "" +
tiempomyql), mysqlBackup);
                runOralce("sh -x /opt/oracle/BACKUP_grande.sh");
                fechaActual = new Date();
                finoracle = formatter.format(fechaActual);
                TreeNode oracleBackupGrande = new
DefaultTreeNode(new Resultados("Grande", finmysql, finoracle, "" +
tiemporacle), oracleBackup);
                pieModel = new PieChartModel();
                if (tiempomyql > tiemporacle) {
                    max = tiempomyql;
                } else {
                    max = tiemporacle;
                }
                pieModel.set("Mysql", tiempomyql);
                pieModel.set("Oracle", tiemporacle);
```

```
                restore = new CartesianChartModel();
                ChartSeries oracle_restore = new ChartSeries();
                oracle_restore.setLabel("Oracle");
                oracle_restore.set("Tiempo (milseg)", tiemporacle);
                ChartSeries mysql_restore = new ChartSeries();
                mysql_restore.setLabel("Mysql");
                mysql_restore.set("Tiempo (milseg)", tiempomyql);
                restore.addSeries(mysql_restore);
                restore.addSeries(oracle_restore);

                LogBase lobBases = new LogBase();
                lobBases.setId(logBaseFacadeLocal.count() + 1);
                lobBases.setFecha(new Date());
                lobBases.setMysql(tiempomyql);
                lobBases.setOracle(tiemporacle);
                lobBases.setTipo("Backup");


                SimpleDateFormat format = new
SimpleDateFormat("ddMMyyyy");
                mostrarLogOracle = "cat
/opt/backups_oracle/archivelogs/log1fullbackup_" + format.format(new
Date()) + "*.log";
```

```
                logBaseFacadeLocal.create(lobBases);
            }
            //Bacukt
            break;
        default:
            break;

    }
    logmysql = logmysql + "Se realizo el respaldo con las siguiente
tablas y registros";
    baseM = new MostrarBases();
    List<Object[]> bases = baseM.mostrarTablas(baseMysqlSeleted);
    List<Object[]> datos = baseM.totalDeTablasYRegistros(bases,
baseMysqlSeleted);
    resul = new ArrayList<Resultados>();
    resulOracle = new ArrayList<Resultados>();
    int totalMysqlTavlas = 0;
    for (Object[] base1 : datos) {
        //resul.add(new Resultados((String)base1[0], "", "",
(String)base1[1]));
        totalMysqlTavlas++;
    }
```

```
            resul.add(new Resultados("Total de Tablas respaldadas Mysql", "",
"", Integer.toString(totalMysqlTavlas)));
            baseOracle = new MostrarBasesOracle();
            List<String> basesoracle =
baseOracle.mostrarTablas(baseOracleSeleted,claveOracle);
            //List<Object[]>
datosOracle=baseOracle.totalDeTablasYRegistros(basesoracle,
baseOracleSeleted);
            //resulOracle = new ArrayList<Resultados>();
            //int totalTablas=0;
            /*for (Object[] base2 : datosOracle) {
             totalTablas++;
             //resulOracle.add(new Resultados((String)base2[0], "", "",
(String)base2[1]));

             }*/

             resulOracle.add(new Resultados("Total de Tablas respaldadas ",
"", "", basesoracle.get(0)));
        } else {
            if (!conexion_mysql.equals("OK") &&
!conexion_oracle.equals("OK")) {
```

```
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error el servidor de
Datos Mysql no esta respondiendo", conexion_mysql));
            logmysql = "Error el servidor de Datos Mysql no esta
respondiendo " + conexion_mysql;
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error el servidor de
Datos Oracle no esta respondiendo", conexion_oracle));
            logoracle = "Error el servidor de Datos Oracle no esta
respondiendo " + conexion_oracle;

        } else {

            if (!conexion_mysql.equals("OK")) {
                FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error el servidor de
Datos Mysql no esta respondiendo", conexion_mysql));
                logmysql = "Error el servidor de Datos Mysql no esta
respondiendo " + conexion_mysql;
            } else {
                FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error el servidor de
Datos Oracle no esta respondiendo", conexion_oracle));
```

```
                logoracle = "Error el servidor de Datos Oracle no esta
respondiendo " + conexion_oracle;
            }
          }
        }
        /*
         * Format formatter; Date fechaActual = new Date(); formatter = new
         * SimpleDateFormat("yyyy/MM/dd.HH:mm:ss");
         *
         * vergrafico = true; iniciomysql = formatter.format(fechaActual);
         * runMYSQL(); fechaActual = new Date(); finmysql =
         * formatter.format(fechaActual); listResult.add(new
Resultados("Mysql",
         * iniciomysql, finmysql, "00")); runOralce(); fechaActual = new Date();
         * finoracle = formatter.format(fechaActual); listResult.add(new
         * Resultados("Oracle", finmysql, finoracle, "00")); //
         * createCategoryModel();
         */
    }

    public List<Resultados> getResulOracle() {
        return resulOracle;
    }
```

```
    public void setResulOracle(List<Resultados> resulOracle) {
        this.resulOracle = resulOracle;
    }

    public void runMYSQL(String comando) {
        String user = PropertyFile.getString("user_mysql");
        String host = PropertyFile.getString("ip_mysql");
        Integer port = 22;
        String pass = PropertyFile.getString("pass_mysql");
        try {
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();

            channelExec.setCommand(comando);
```

```
            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            int index = 0;

            while ((linea = reader.readLine()) != null) {
                logmysql = logmysql + (++index + " : " + linea + "\n");
                System.out.println(++index + " : " + linea);
            }

            channelExec.disconnect();
            session.disconnect();

            //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");
            tiempomyql = ((System.currentTimeMillis() - s));
        } catch (JSchException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
```

```
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }

    }

    public void handleCityChange() {
        logmysql = "";
        logoracle = "";
        pieModel = new PieChartModel();
//                  tiempomyql = 10;
//                  tiemporacle = 30;
        max = 0;
        pieModel.set("Mysql", tiempomyql);
        pieModel.set("Oracle", tiemporacle);
        restore = new CartesianChartModel();
        ChartSeries oracle_restore = new ChartSeries();
        oracle_restore.setLabel("Oracle");
        oracle_restore.set("Tiempo (milseg)", 0);
        ChartSeries mysql_restore = new ChartSeries();
        mysql_restore.setLabel("Mysql");
```

```
        mysql_restore.set("Tiempo (milseg)", 0);
        restore.addSeries(mysql_restore);
        restore.addSeries(oracle_restore);
    }

    public void runOralce(String comando) {
        String user = PropertyFile.getString("user_oracle");
        String host = PropertyFile.getString("ip_oracle");
        Integer port = 22;
        String pass = PropertyFile.getString("pass_oracle");
         long s = System.currentTimeMillis();
        try {

            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();
```

```
            channelExec.setCommand(comando);

            /*
             * channelExec.setCommand("cd /home/icevallos/");
             * channelExec.setCommand("ls");
             */
            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            int index = 0;

            while ((linea = reader.readLine()) != null) {
               logoracle = logoracle + (++index + " : " + linea + "\n");
               System.out.println(++index + " : " + linea);
            }

            channelExec.disconnect();
            session.disconnect();

            //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");
```

```
            tiemporacle = ((System.currentTimeMillis() - s));
        } catch (JSchException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }

    }

    public CartesianChartModel getRestore() {
        return restore;
    }
//
//    public CartesianChartModel getBackup() {
//        return backup;
//    }

    private void verGrafico() {
        restore = new CartesianChartModel();
```

```
    ChartSeries oracle_restore = new ChartSeries();
    oracle_restore.setLabel("Oracle");
    oracle_restore.set("Tiempo", 0);
    ChartSeries mysql_restore = new ChartSeries();
    mysql_restore.setLabel("Mysql");
    mysql_restore.set("Tiempo", 0);
    restore.addSeries(mysql_restore);
    restore.addSeries(oracle_restore);
}

public String getLogmysql() {
    return logmysql;
}

public void setLogmysql(String logmysql) {
    this.logmysql = logmysql;
}

public String getLogoracle() {
    return logoracle;
}

public void setLogoracle(String logoracle) {
```

```
    this.logoracle = logoracle;
}
public long getTiempomyql() {
    return tiempomyql;
}

public void setTiempomyql(long tiempomyql) {
    this.tiempomyql = tiempomyql;
}

public long getTiemporacle() {
    return tiemporacle;
}

public void setTiemporacle(long tiemporacle) {
    this.tiemporacle = tiemporacle;
}

public boolean isVergrafico() {
    return vergrafico;
}

public void setVergrafico(boolean vergrafico) {
```

```
    this.vergrafico = vergrafico;
}

public List<Resultados> getListResult() {
    return listResult;
}

public void setListResult(List<Resultados> listResult) {
    this.listResult = listResult;
}

public String getBase() {
    return base;
}

public void setBase(String base) {
    this.base = base;
}

public int getTipo() {
    return tipo;
}
```

```
public void setTipo(int tipo) {
    this.tipo = tipo;
}

public TreeNode getRoot() {
    return root;
}

public List<Recurso> getCaracteristicas() {
    return caracteristicas;
}

public void setCaracteristicas(List<Recurso> caracteristicas) {
    this.caracteristicas = caracteristicas;
}

public String getLog_mysql() {
    return log_mysql;
}

public void setLog_mysql(String log_mysql) {
    this.log_mysql = log_mysql;
}
```

```
    private void getMemoriaMyql() {
        String user = PropertyFile.getString("user_mysql");
        String host = PropertyFile.getString("ip_mysql");
        Integer port = 22;
        String pass = PropertyFile.getString("pass_mysql");
        try {
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();

            channelExec.setCommand("sh -x /opt/propiedades.sh");
            //channelExec.setCommand("cat /proc/meminfo");
            /*
             *
             * channelExec.setCommand("ls");
```

```
             */
            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            String data;

            while ((linea = reader.readLine()) != null) {
               //System.out.println(linea);
               if (linea.indexOf(":") > 0) {
                  data = linea.substring(0, linea.indexOf(":")).trim();
                  System.out.println(linea.length());
                  if (data.equals("model name")) {
                     procesador_mysql = linea.substring(linea.indexOf(":") + 1,
linea.length());

                  } else if (data.equals("MemTotal")) {
                     memoria_mysql = linea.substring(linea.indexOf(":") + 1,
linea.length());
                     System.out.println(linea.length());
                  }
               }
```

```
                if (linea.indexOf("dm-0") > 0) {
                    disco_mysql = linea.substring(linea.indexOf("/dev/dm-0"),
linea.length()).trim();
                }

            }
            channelExec.disconnect();
            session.disconnect();

        } catch (JSchException e) {
            System.out.println(e.getMessage());

        } catch (IOException e) {
            System.out.println(e.getMessage());
        }

    }

    private void getMemoriaOracle() {
        String user = PropertyFile.getString("user_oracle");
        String host = PropertyFile.getString("ip_oracle");
        Integer port = 22;
        String pass = PropertyFile.getString("pass_oracle");
```

```
        try {
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");
            InputStream in = channelExec.getInputStream();
            channelExec.setCommand("sh -x /opt/propiedades.sh");
            //channelExec.setCommand("cat /proc/meminfo");
            /*
             * channelExec.setCommand("cd /home/jronquillo/");
             * channelExec.setCommand("ls");
             */
            channelExec.connect();
            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            String data;
            while ((linea = reader.readLine()) != null) {
```

```
                if (linea.indexOf(":") > 0) {
                    data = linea.substring(0, linea.indexOf(":")).trim();
                    if (data.equals("model name")) {
                        procesador_oracle = linea.substring(linea.indexOf(":") + 1,
linea.length());
                    } else if (data.equals("MemTotal")) {
                        memoria_oracle = linea.substring(linea.indexOf(":") + 1,
linea.length());
                    }
                }
                if (linea.indexOf("sda8") > 0) {
                    disco_oracle = linea.substring(linea.indexOf("/dev/sda8"),
linea.length()).trim();
                }
            }
            channelExec.disconnect();
            session.disconnect();
        } catch (JSchException e) {
            System.out.println(e.getMessage());
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
    }
```

```
    public void estado() {
        String user = PropertyFile.getString("user_oracle");
        String host = PropertyFile.getString("ip_oracle");
        Integer port = 22;
        String pass = PropertyFile.getString("pass_oracle");
        try {
//            System.out.println("1");
            // long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");
            InputStream in = channelExec.getInputStream();
            channelExec.setCommand("top -bn 1");
            /*
             * channelExec.setCommand("cd /home/icevallos/");
             * channelExec.setCommand("ls");
             */
            channelExec.connect();
```

```
        BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
        String linea;
        //String data;
        int i = 0;
        while ((linea = reader.readLine()) != null) {
            ++i;
            if (i >= 2 && i <= 6) {
                switch (i) {
                    case 2:
                        log.get(0).setMysql(linea);
                        break;
                    case 3:
                        log.get(1).setMysql(linea);
                        break;
                    case 4:
                        log.get(2).setMysql(linea);
                        break;
                    case 5:
                        log.get(3).setMysql(linea);
                        break;

                }
```

```
            }
        }
        channelExec.disconnect();
        session.disconnect();

    } catch (JSchException e) {
        System.out.println(e.getMessage());

    } catch (IOException e) {
        System.out.println(e.getMessage());
    }

    String user_mysql = PropertyFile.getString("user_mysql");
    String host_mysql = PropertyFile.getString("ip_mysql");
    Integer port_mysql = 22;
    String pass_mysql = PropertyFile.getString("pass_mysql");
    try {
        // long s = System.currentTimeMillis();
        JSch jsch = new JSch();
        Session session = jsch.getSession(user_mysql, host_mysql,
port_mysql);
        UserInfo ui = new SUserInfo(pass_mysql, null);
        session.setUserInfo(ui);
```

```
            session.setPassword(pass_mysql);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();

            channelExec.setCommand("top -bn 1");

            /*
             * channelExec.setCommand("cd /home/icevallos/");
             * channelExec.setCommand("ls");
             */
            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;

            int i = 0;

            while ((linea = reader.readLine()) != null) {
                ++i;
```

```
    if (i >= 2 && i <= 6) {
        switch (i) {
            case 2:
                log.get(0).setOracle(linea);
                break;

            case 3:
                log.get(1).setOracle(linea);
                break;
            case 4:
                log.get(2).setOracle(linea);
                break;
            case 5:
                log.get(3).setOracle(linea);
                break;

        }

    }
}
channelExec.disconnect();
session.disconnect();
```

```
    } catch (JSchException e) {
        System.out.println(e.getMessage());

    } catch (IOException e) {
        System.out.println(e.getMessage());
    }

}

public String getTituloGrafico() {
    return tituloGrafico;
}

public void setTituloGrafico(String tituloGrafico) {
    this.tituloGrafico = tituloGrafico;
}

public List<Recurso> getLog() {
    return log;
}

public void setLog(List<Recurso> log) {
    this.log = log;
```

```
}

public long getMax() {
    return max;
}

public void setMax(long max) {
    this.max = max;
}

public LogBase getLogbase() {
    return logbase;
}

public void setLogbase(LogBase logbase) {
    this.logbase = logbase;
}

public List<LogBase> getListLogBase() {
    return logBaseFacadeLocal.findAll();
}

public void setListLogBase(List<LogBase> listLogBase) {
```

```
        this.listLogBase = listLogBase;
    }

    public void mostrarGrafico(LogBase l){
        grafico = new PieChartModel();
        grafico.set("Mysql", l.getMysql());
       grafico.set("Oracle", l.getOracle());
    }
public PieChartModel getGrafico() {
        grafico = new PieChartModel();
        if (logbase != null) {
            grafico.set("Mysql", logbase.getMysql());
            grafico.set("Oracle", logbase.getOracle());
        }else{
           grafico.set("Mysql", 0);
            grafico.set("Oracle", 0);
        }
        return grafico;
    }
    public void setGrafico(PieChartModel grafico) {
        this.grafico = grafico;
    }
```

```
public List<Resultados> getResul() {
    return resul;
}

public void setResul(List<Resultados> resul) {
    this.resul = resul;
}

public String getResult() {

    String resultado = "";
    String user = PropertyFile.getString("user_oracle");
    String host = PropertyFile.getString("ip_oracle");
    Integer port = 22;
    String pass = PropertyFile.getString("pass_oracle");
    try {
        long s = System.currentTimeMillis();
        JSch jsch = new JSch();
        Session session = jsch.getSession(user, host, port);
        UserInfo ui = new SUserInfo(pass, null);
        session.setUserInfo(ui);
        session.setPassword(pass);
        session.connect();
```

```
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();



            String logNombre = mostrarLogOracle;
            channelExec.setCommand(logNombre);
            System.out.println(logNombre);
            /*
             * channelExec.setCommand("cd /home/icevallos/");
             * channelExec.setCommand("ls");
             */
            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            int index = 0;

            while ((linea = reader.readLine()) != null) {
                resultado = resultado + linea + "\n";
                System.out.println(++index + " : " + linea);
```

```
            }

            channelExec.disconnect();
            session.disconnect();

            //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");
            tiemporacle = ((System.currentTimeMillis() - s));
        } catch (JSchException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }
        return resultado;
    }

    public void inicarservicio() {
        String resultado = "";
        String user = PropertyFile.getString("user_mysql");
```

```
        String host = PropertyFile.getString("ip_mysql");
        Integer port = 22;
        String pass = PropertyFile.getString("pass_mysql");
        try {
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();

            channelExec.setCommand("service mysqld start");

            /*
             * channelExec.setCommand("cd /home/jronquillo/");
             * channelExec.setCommand("ls");
             */
            channelExec.connect();
```

```
            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            int index = 0;

            while ((linea = reader.readLine()) != null) {
                resultado = resultado + linea + "\n";
                FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_INFO, "Mensaje", "El servicio
de base datos fue iniciado Mysql: " + linea));
                System.out.println(++index + " : " + linea);
            }

            channelExec.disconnect();
            session.disconnect();

            //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");
            // tiemporacle = ((System.currentTimeMillis() - s) );
        } catch (JSchException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
```

```
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }

        user = PropertyFile.getString("user_oracle");
        host = PropertyFile.getString("ip_oracle");
        port = 22;
        pass = PropertyFile.getString("pass_oracle");

        try {
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();
```

```
            channelExec.setCommand("sh /opt/levanta_sid.sh");

            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            int index = 0;

            while ((linea = reader.readLine()) != null) {
               resultado = resultado + linea + "\n";
               FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_INFO, "Mensaje", "El servicio
de base datos fue iniciado Oracle: " + linea));
            }

            channelExec.disconnect();
            session.disconnect();

            //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");
            //tiemporacle = ((System.currentTimeMillis() - s) );
        } catch (JSchException e) {
```

```
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }

    }

    public void reinicarservicio() {
        String resultado = "";
        String user = PropertyFile.getString("user_mysql");
        String host = PropertyFile.getString("ip_mysql");
        Integer port = 22;
        String pass = PropertyFile.getString("pass_mysql");
        try {
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
```

```
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();
            /*
             * channelExec.setCommand("cd /home/icevallos/");
             * channelExec.setCommand("ls");
             */
            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            int index = 0;

            while ((linea = reader.readLine()) != null) {
                resultado = resultado + linea + "\n";
                channelExec.setCommand("service mysqld restart ");
                FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_INFO, "Mensaje", "El servicio
de base datos fue reinicado" + linea));
```

```
            }

            channelExec.disconnect();
            session.disconnect();

            //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");
            //  tiemporacle = ((System.currentTimeMillis() - s) );
        } catch (JSchException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }

        user = PropertyFile.getString("user_oracle");
        host = PropertyFile.getString("ip_oracle");
        port = 22;
        pass = PropertyFile.getString("pass_oracle");
        try {
```

```
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();

            channelExec.setCommand("sh /opt/reinicia_sid.sh");

            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            int index = 0;

            while ((linea = reader.readLine()) != null) {
                resultado = resultado + linea + "\n";
```

```
                FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_INFO, "Mensaje", "El servicio
de base datos fue reiniciado Oracle: " + linea));
            }

            channelExec.disconnect();
            session.disconnect();

            //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");
            //tiemporacle = ((System.currentTimeMillis() - s) );
        } catch (JSchException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }

    }
```

```
    public void parar() {
        String resultado = "";
        String user = PropertyFile.getString("user_mysql");
        String host = PropertyFile.getString("ip_mysql");
        Integer port = 22;
        String pass = PropertyFile.getString("pass_mysql");
        try {
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");

            InputStream in = channelExec.getInputStream();

            channelExec.setCommand("service mysqld stop");

            /*
             * channelExec.setCommand("cd /home/icevallos/");
```

```
             * channelExec.setCommand("ls");
             */
            channelExec.connect();

            BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
            String linea;
            int index = 0;

            while ((linea = reader.readLine()) != null) {
                resultado = resultado + linea + "\n";
                FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_INFO, "Mensaje", "El servicio
de base datos fue parado" + linea));
            }

            channelExec.disconnect();
            session.disconnect();

            //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");

        } catch (JSchException e) {
```

```
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }

        user = PropertyFile.getString("user_oracle");
        host = PropertyFile.getString("ip_oracle");
        port = 22;
        pass = PropertyFile.getString("pass_oracle");
        try {
            long s = System.currentTimeMillis();
            JSch jsch = new JSch();
            Session session = jsch.getSession(user, host, port);
            UserInfo ui = new SUserInfo(pass, null);
            session.setUserInfo(ui);
            session.setPassword(pass);
            session.connect();
            ChannelExec channelExec = (ChannelExec)
session.openChannel("exec");
```

```
        InputStream in = channelExec.getInputStream();

        channelExec.setCommand("sh /opt/parar_sid.sh");

        channelExec.connect();

        BufferedReader reader = new BufferedReader(new
InputStreamReader(in));
        String linea;
        int index = 0;

        while ((linea = reader.readLine()) != null) {
            resultado = resultado + linea + "\n";
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_INFO, "Mensaje", "El servicio
de base datos fue iniciado Oracle: " + linea));
        }

        channelExec.disconnect();
        session.disconnect();

        //System.out.println("Tiempo: " + (System.currentTimeMillis() - s) /
1000 + " ms. ");
```

```
            //tiemporacle = ((System.currentTimeMillis() - s) );
        } catch (JSchException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        } catch (IOException e) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage(FacesMessage.SEVERITY_ERROR, "Error",
e.getMessage()));
        }

    }

    public List<Object[]> getTablaBase() {
        MostrarBases m =new MostrarBases();
        return m.mostrarTablas(baseMysqlSeleted);
    }

    public void setTablaBase(List<Object[]> tablaBase) {
        this.tablaBase = tablaBase;
    }
```

```
    public void buscarBasesmysql(){
        MostrarBases m =new MostrarBases();
       this.tablaBase = m.mostrarTablas(baseMysqlSeleted);
    }

    public List<Object[]> getTablaoracle() {
        MostrarBasesOracle m = new MostrarBasesOracle();
        return m.mostrarTodasTablas(baseOracleSeleted, claveOracle);
    }

    public void setTablaoracle(List<Object[]> tablaoracle) {
        this.tablaoracle = tablaoracle;
    }

    public void buscarBasesoracle(){
        MostrarBasesOracle m = new MostrarBasesOracle();
       tablaoracle=  m.mostrarTodasTablas(baseOracleSeleted,
claveOracle);
    }

}
```

# MANUAL DE USUARIO

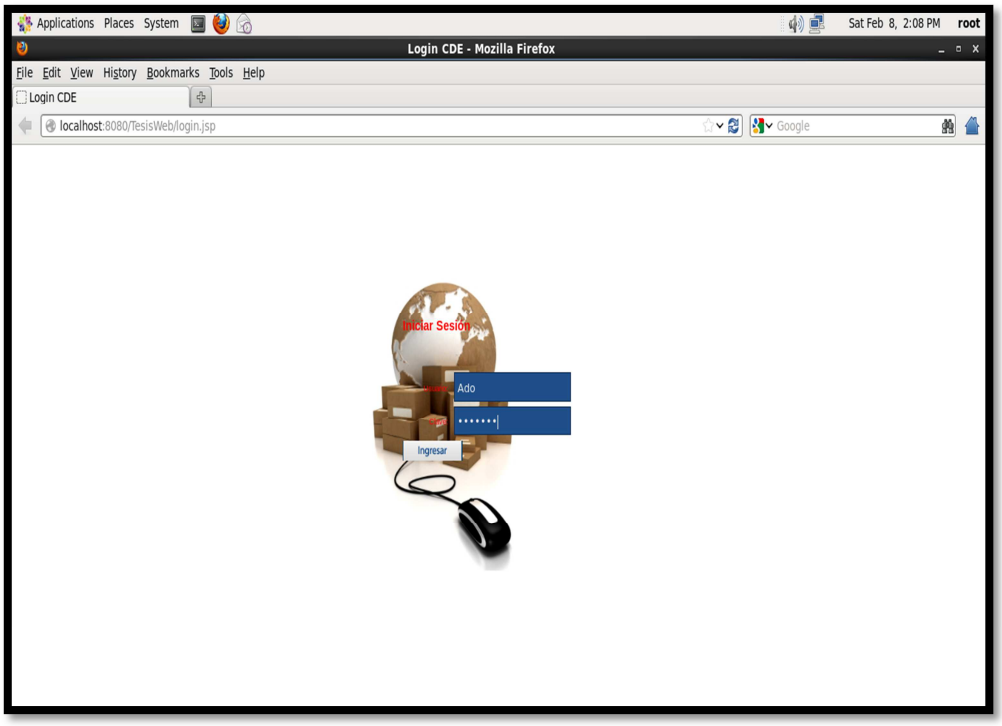
## Introducción

El objetivo de este manual es permitir que la aplicación del Análisis Comparativo de Respaldo y Recuperación de las bases de datos, pueda brindarles a los usuarios una interacción amigable y fácil de usar.

# APLICACIÓN EN NETBEANS IDE 7.2

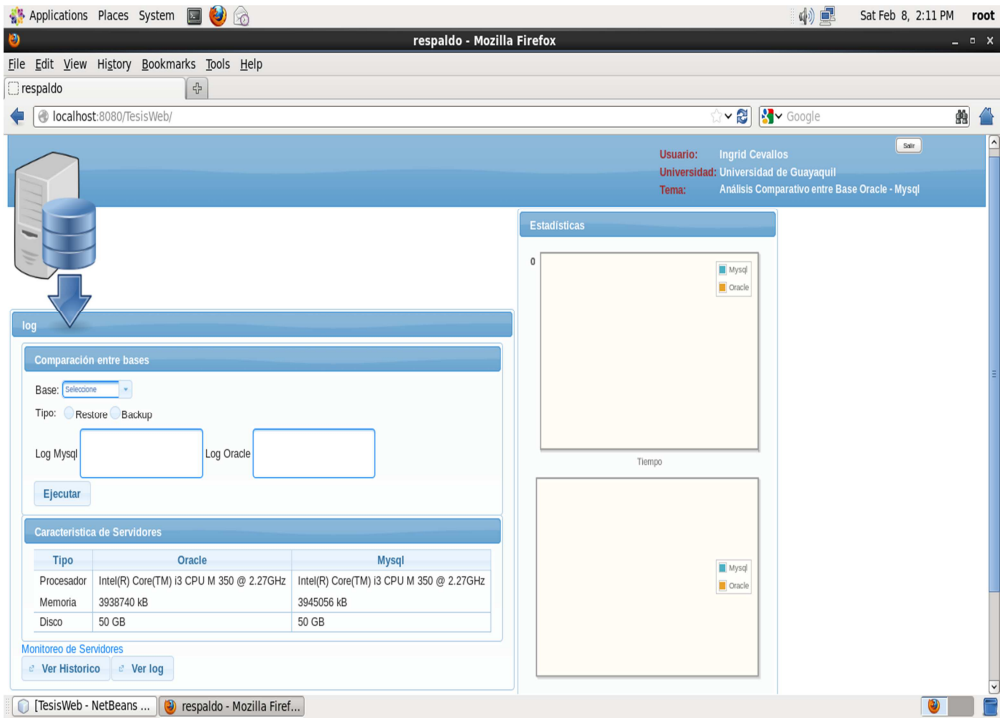
En esta plataforma se ha creado una aplicación en java para establecer y determinar de manera estadística el rendimiento de cada una de las bases, Oracle y Mysql, mientras realiza los procesos de respaldo y recuperación respectivamente con sus herramientas Rman y Mysql Administrator o Mysqldump.

En primer lugar ingreso con mi usuario y contraseña como muestra la pantalla de inicio de sesión, el cual me permite autenticarme como usuario a la aplicación.




**Inicio de Sesión**

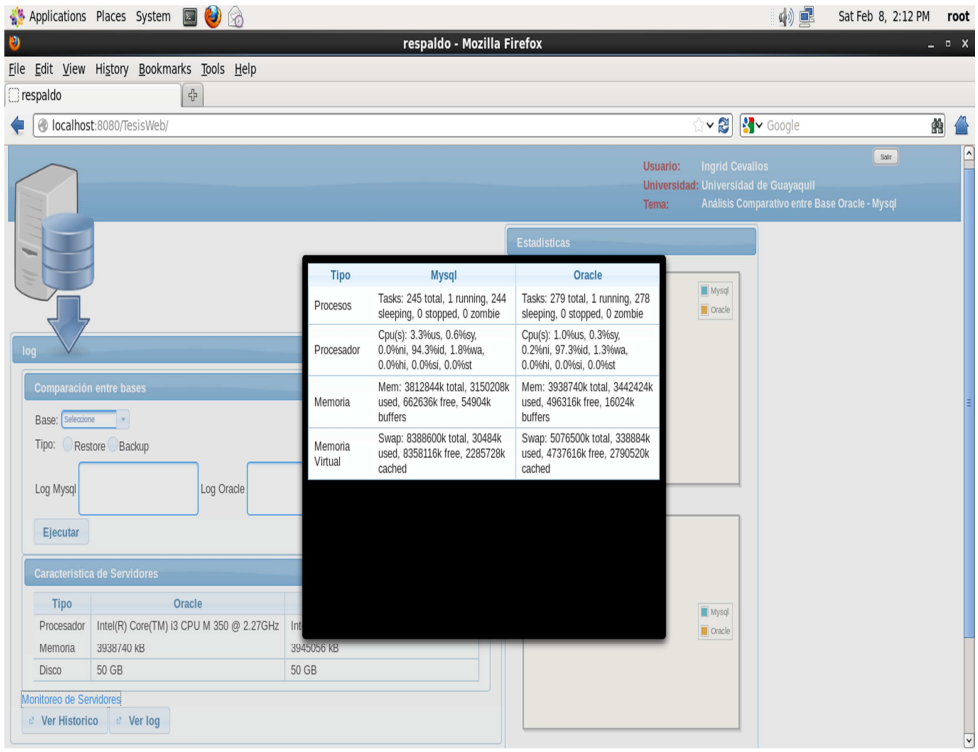
Luego de ingresar a la aplicación, se muestra está pantalla con las opciones:




**Ventana Principal**

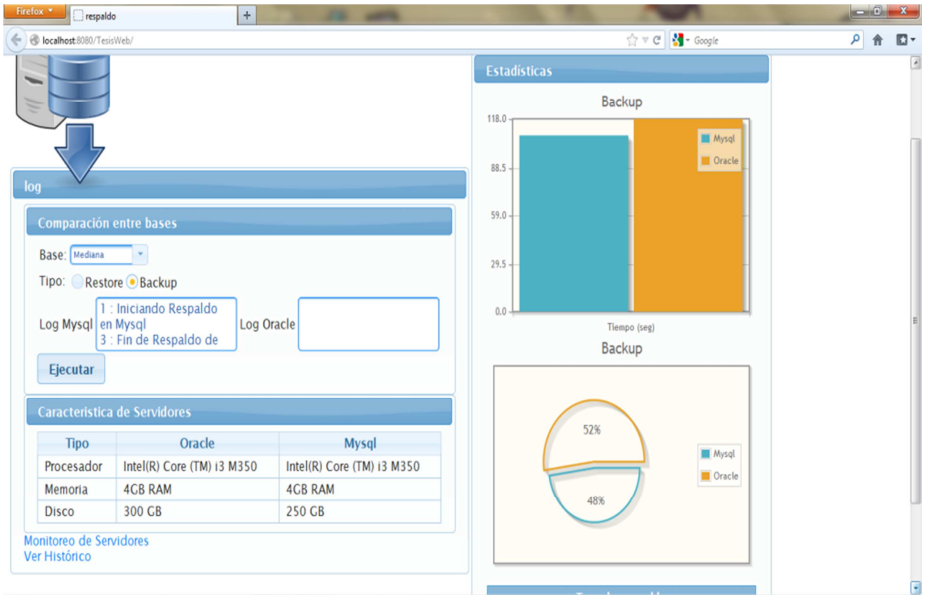
- Base a seleccionar: grande, mediana, pequeña
- Tipo de base, es decir, si es un Backup o un Restore.
- Botón 'Ejecutar', que es donde llamo el shell del backup y restore.
- Características de Servidores, muestra en la pantalla las características de los equipos donde se está realizando el backup y la recuperación, Oracle y Mysql. **(Ventana Principal).**
- Monitoreo de Servidores, donde se va a mostrar el rendimiento del Procesador, memoria y disco, cada vez que se ejecute el respaldo

y la recuperación de cada una de las bases de datos. **(Monitoreo de Servidores).**


Applications Places System
Sat Feb 8, 2:12 PM root
respaldo - Mozilla Firefox
File Edit View History Bookmarks Tools Help
respaldo
localhost:8080/TesisWeb/
Google
Usuario: Ingrid Cevallos
Universidad: Universidad de Guayaquil
Tema: Análisis Comparativo entre Base Oracle - Mysql
Salir
Estadísticas
log
Comparación entre bases
Base: Seleccione
Tipo: Restore Backup
Log Mysql
Log Oracle
Ejecutar
Caracteristica de Servidores
Tipo Oracle
Procesador Intel(R) Core(TM) i3 CPU M 350 @ 2.27GHz
Memoria 3938740 kB
Disco 50 GB 50 GB
Monitoreo de Servidores
Ver Historico
Ver log
Tipo Mysql Oracle
Procesos Tasks: 245 total, 1 running, 244 sleeping, 0 stopped, 0 zombie Tasks: 279 total, 1 running, 278 sleeping, 0 stopped, 0 zombie
Procesador Cpu(s): 3.3%us, 0.6%sy, 0.0%ni, 94.3%id, 1.8%wa, 0.0%hi, 0.0%si, 0.0%st Cpu(s): 1.0%us, 0.3%sy, 0.2%ni, 97.3%id, 1.3%wa, 0.0%hi, 0.0%si, 0.0%st
Memoria Mem: 3812844k total, 3150208k used, 662636k free, 54904k buffers Mem: 3938740k total, 3442424k used, 496316k free, 16024k buffers
Memoria Virtual Swap: 8388600k total, 30484k used, 8358116k free, 2285728k cached Swap: 5076500k total, 338884k used, 4737616k free, 2790520k cached

**Monitoreo de Servidores**

- Botón para iniciar, parar y reiniciar la base de datos.
- En el lado derecho de la aplicación muestra automáticamente las estadísticas por tiempo y porcentaje cuando se realiza el respaldo y el restore.




**Estadísticas de respaldo y restore**

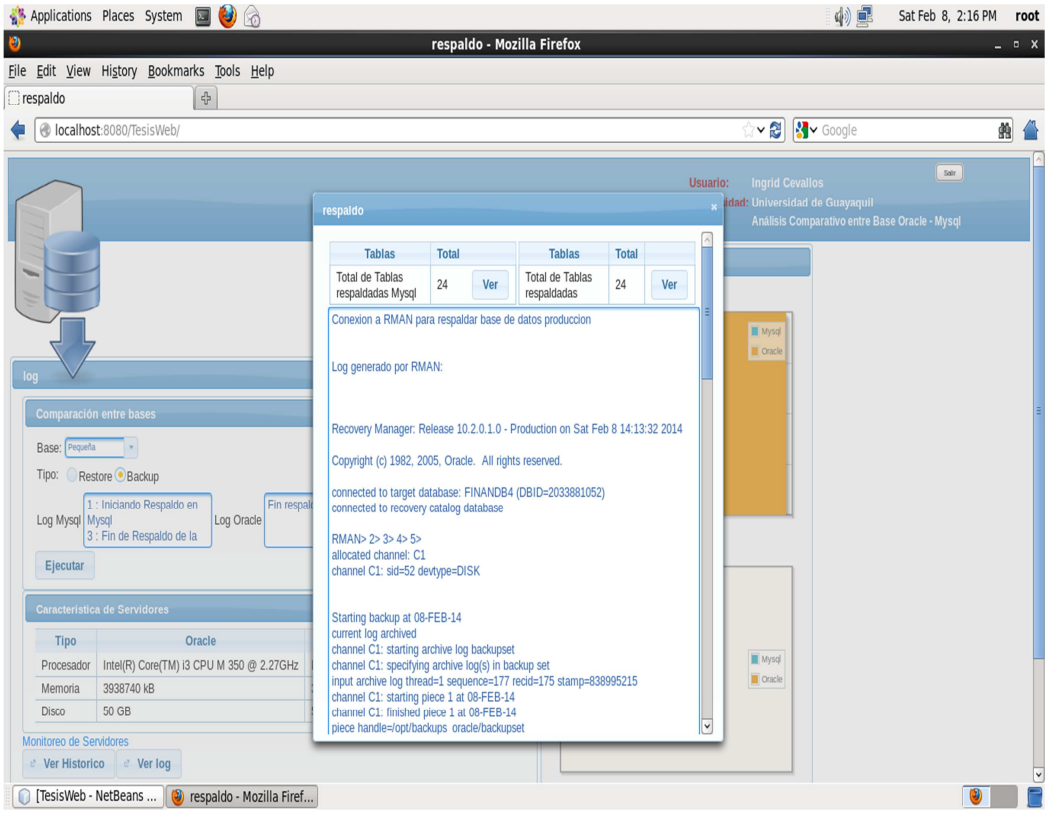
- **Ver Histórico**, en esta opción me va a mostrar el registro de todo lo procesado, la fecha, tiempo y el histórico acompañado de un gráfico.

| Tipo | Fecha | Tiempo Mysql | Tiempo Oracle |
|---|---|---|---|
| Restore | 07/10/2012 | 10 | 30 |
| Restore | 07/10/2012 | 20 | 50 |
| Restore | 22/10/2012 | 0 | 0 |
| Backup | 22/10/2012 | 0 | 0 |
| Backup | 22/10/2012 | 15 | 0 |
| Backup | 22/10/2012 | 12 | 0 |
| Backup | 22/10/2012 | 11 | 0 |
| Backup | 22/10/2012 | 66 | 0 |
| Backup | 22/10/2012 | 56 | 0 |
| Backup | 22/10/2012 | 54 | 135 |
| Backup | 22/10/2012 | 73 | 150 |
| Restore | 22/10/2012 | 0 | 0 |
| Backup | 23/10/2012 | 108 | 118 |

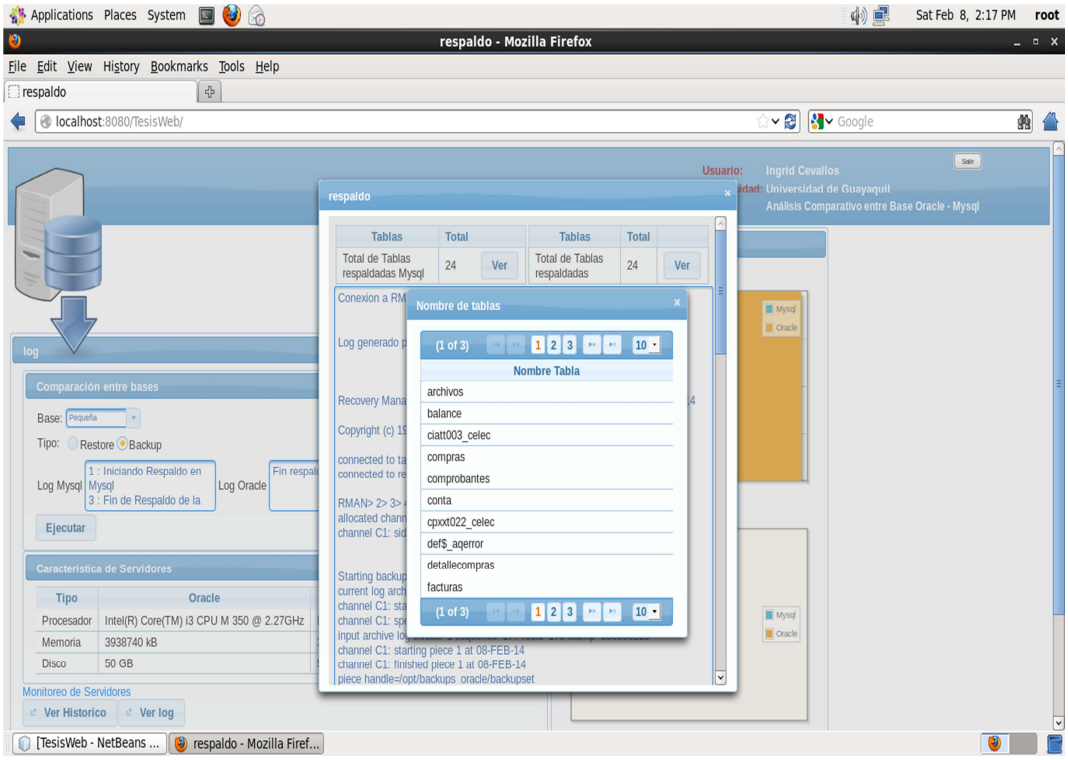
**Histórico de respaldo y restore**

**Ver log,** en esta opción se puede observar las tablas y el total de tablas respaldadas de las dos bases de datos tanto oracle como mysql. **(Log_Tablas)**

Además del log que muestra cuando se está realizando el respaldo en oracle, mysql no muestra log.


Applications Places System
Sat Feb 8, 2:16 PM root
respaldo - Mozilla Firefox
File Edit View History Bookmarks Tools Help
respaldo
localhost:8080/TesisWeb/
Usuario: Ingrid Cevallos
Universidad de Guayaquil
Análisis Comparativo entre Base Oracle - Mysql
respaldo
Tablas Total Tablas Total
Total de Tablas respaldadas Mysql 24 Ver
Total de Tablas respaldadas 24 Ver
Conexion a RMAN para respaldar base de datos produccion
Log generado por RMAN:
Recovery Manager: Release 10.2.0.1.0 - Production on Sat Feb 8 14:13:32 2014
Copyright (c) 1982, 2005, Oracle. All rights reserved.
connected to target database: FINANDB4 (DBID=2033881052)
connected to recovery catalog database
RMAN> 2> 3> 4> 5>
allocated channel: C1
channel C1: sid=52 devtype=DISK
Starting backup at 08-FEB-14
current log archived
channel C1: starting archive log backupset
channel C1: specifying archive log(s) in backup set
input archive log thread=1 sequence=177 recid=175 stamp=838995215
channel C1: starting piece 1 at 08-FEB-14
channel C1: finished piece 1 at 08-FEB-14
piece handle=/opt/backups oracle/backupset
log
Comparación entre bases
Base: Pequeña
Tipo: Restore Backup
Log Mysql
1 : Iniciando Respaldo en Mysql
3 : Fin de Respaldo de la
Log Oracle
Ejecutar
Caracteristica de Servidores
Tipo Oracle
Procesador Intel(R) Core(TM) i3 CPU M 350 @ 2.27GHz
Memoria 3938740 kB
Disco 50 GB
Monitoreo de Servidores
Ver Historico
Ver log
[TesisWeb - NetBeans ...
respaldo - Mozilla Firef...

**Log**

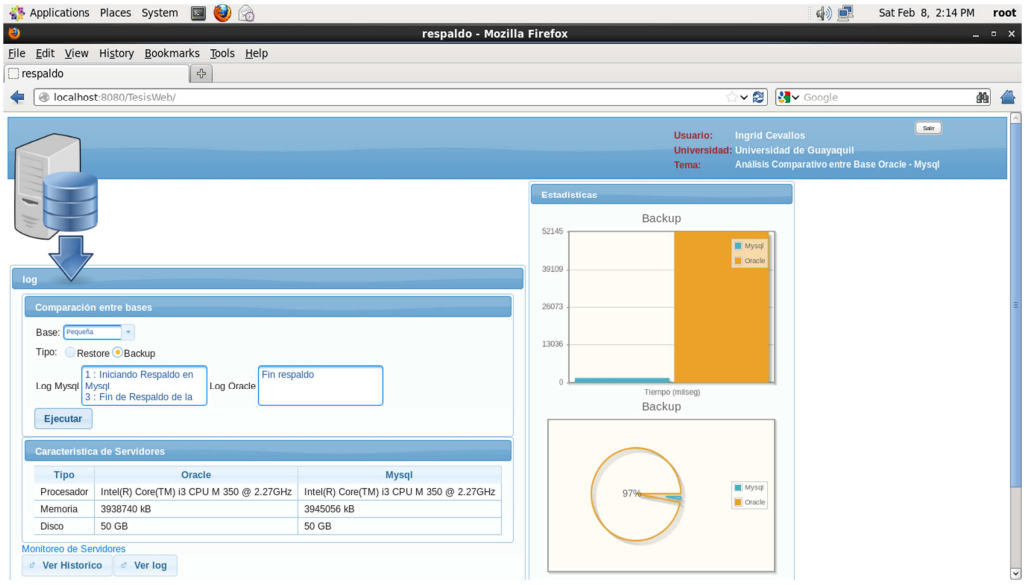



**Log_tablas**

- **Respaldo de base de datos pequeña**

  Para realizar un respaldo de una base de datos pequeña, debemos seleccionar la opción **Base** y **Tipo,** en este caso **Backup** y luego damos clic sobre el botón **Ejecutar.**

  Luego de ejecutarse el respaldo de la base de datos, la aplicación me mostrará en forma estadística y en porcentaje el tiempo de respuesta del respaldo.

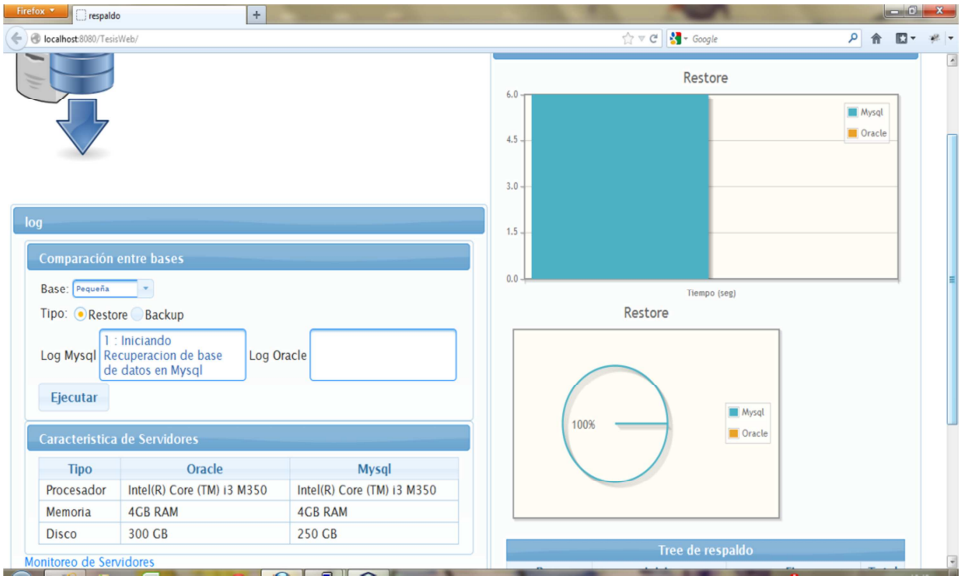



**Nota:** Este paso se debe realizar para la base que deseemos sacar el respaldo.

- **Recuperación de base de datos pequeña**

  Para realizar la recuperacion de una base de datos pequeña, debemos seleccionar la opción **Base** y **Tipo**, en este caso **Restore** y luego damos clic sobre el botón **Ejecutar.**

  Luego de ejecutarse la recuperación de la base de datos, la aplicación me mostrará en forma estadística y en porcentaje el tiempo de respuesta de la recuperación.




**Nota:** Este paso se debe realizar para la base que deseemos sacar el respaldo.